Tid / celle	Før start	1. ord indlæst og lagret	2. ord indlæst og lagret	3. ord indlæst og lagret	4. ord indlæst og lagret
34	pm n 64 DXIZA	pm n 64 DXIZA	pm n 64 DXIZA	pm n 64 DXIZA	pm n 64 DXIZA
35	tl -7 , ly r+1	tl -7 , ly r+1	tl -7 , ly r+1	tl -7 , ly r+1	tl -7 , ly r+1
36	pi 0 LZA t508	pi 0 LZA t508	pi 0 LZA t508	pi 0 LZA t508	pi 0 LZA t508
37	xr 0 XIZB	xr 0 XIZB	xr 0 XIZB	xr 0 XIZB	xr 0 XIZB
38	hv 35 LZB	hv 35 LZB	hv 35 LZB	hv 35 LZB	hv 35 LZB
39	gr 41 MRC t-1	gr 40 MRC t-1	gr 42 MRC t-1	gr 41 MRC t-1	gr 40 MRC t-1
40		hv 34	hv 34	hv 34	gr 46 MRC t-2
41				hv 34	hv 34
42					
43					
44					
45					
46					

Tid / celle	4. ord indlæst og lagret	5. ord indlæst og lagret	5. ord indlæst og lagret	6. ord indlæst og lagret	Sidste ord indlæst og lagret
34	pm n 64 DXIZA	pm n 64 DXIZA	pm n 64 DXIZA	pm n 64 DXIZA	
35	tl -7 , ly r+1	tl -7 , ly r+1	tl -7 , ly r+1	tl -7 , ly r+1	
36	pi 0 LZA t508	pi 0 LZA t508	pi 0 LZA t508	pi 0 LZA t508	
37	xr 0 XIZB	xr 0 XIZB	xr 0 XIZB	xr XIZB	
38	hv 35 LZB	hv 35 LZB	hv 35 LZB	hv 35 LZB	
39	gr 40 MRC t-1	tl 3	tl 3	tl 3	
40	gr 44 MRC t-2	gr 44 MRC t-2	gr 42 MRC t-2	gr i MRC t-1	gr 41 MRC t-1
41	hv 34	hv 34	hv 34	hv 34	
42			tl 3	tl 3	
43					
44	gr 46 MRC t-2	gr 46 MRC t-2	gr 46 MRC t-2	gr 46 MRC t-2	
45					Indlæste program
46					

REGNECENTRALEN
Dansk Institut for Matematikmaskiner

Tegnet
Kontrol
Godk.

10.8.63.

Startprocedure ved indlæsning med kanal 0

Testprograms Type C
RIALTO, 19.8.1968
JFM

The usual testprograms of B type can not be read by the primitive input program on track 0 in HELP 3. This paper describes how the modified tapes, called C programs, may be read by HELP 3.

FORMAT OF C TAPES

The paper tapes containing C programs consists of a short header followed by the corresponding B program. The header can be read by HELP 3 and enables the user to read the B program.

INPUT PROCEDURE

The header is followed by a tape gap 1" long. The reading may start before the header or in this gap. If the should be read by HJÆLP one starts in the gap, and if HELP 3 is used for reading one starts before the header. In the last case the input procedure is as follows:

(1) Read track 0 to the core store by
   a. Pressing RESET or MICROTEMPI STOP.
   b. Pressing HP-BUTTON while NORMAL STOP is pressed.

(2) Place the tape in the reader and start GIER in cell 23 by
   a. Setting r1=23
   b. BY=0
   c. Pressing NORMAL START

(3) When the program types
   t < number of program >
   the header hes been read and the remaining part of the tape (the B program) will be read immidiately.

(4) When the B program has been read, the program in question starts running.

The actual procedure between (2) and (3) above is:

(A) A program (in bin 0 form) containing the primitive loader of HJÆLP is read to the core store in cells 493-512 and entered.

(B) This program types the test number and moves the loader to core 34-39.

(C) The program jumps to core 34 and from this point the action is as if track 0 of HJÆLP has been called.

PUNCHING OF C PROGRAMS

C programs are punched by the slip program A-224, "B to C Producer", and this program and how to use it is described in a seperate paper.

[Testprogram no. 1.Topsøes test of the drum.
The program writes pseudo- random numbers on the tracks 1-319(the same number on all tracks) and compares the number that has been read with the written number.After each comparison for all the tracks is typed ok on the typewriter. By errors is typed track number,word number, the written word and the word that has been read.Bit 40 and 41 do not contain random numbers. KA=1 effects simultaneous computation by writing, KB=1 effects simultaneous computation by reading.Output on the typewriter can be removed by setting by=0 after the testprogram is read in.First instruction to be executed is zqLKB.
The drum parity check should be removed during run by the switch in GIER.]

```
i=41
[  41] zqLKB                    ;stop if KB=1
[  42] vy  16                   ;choice of typewriter
[  43] pa r8 t0                 ;choice of start track
[  44] pa r3 t-305              ;choice of start address for random  numbers
[  45] hs r31                   ;jump to sequ. for random  numbers
[  46] tk 1 ,it 1               ;shift
[  47] gr -305,hs r29           ;store random  numbers from 720 and jump
[  48] tk -30,ac (r-1)          ;shift and add
[  49] it(r-2),bs -265          ;is repeated 40-
[  50] hh r-5                   ;times
[  51] vk 0 t1                  ;choice of track
[  52] sk -304                  ;write track
[  53] hv r5 NKA                ;jump if KA=0
[  54] pa r2 t50                ;put in 50 as counting number
[  55] hs r21 t29               ;compute 50-
[  56] bt 50 t-1                ;random  numbers simultaneously-
[  57] hv r-2                   ;with writing
[  58] vk(r-7),lk -504          ;choose track and read until 520
[  59] hv r5 NKB                ;jump if KB=0
[  60] pa r2 t50                ;place 50
[  61] hs r15                   ;compute 50-
[  62] bt 50 t-1                ;random  numbers simultaneously-
[  63] hv r-2                   ;with reading
[  64] pa r3 t-305              ;choice of start address for random  numbers
[  65] pa r3 t-505              ;choice of start address for read numbers
[  66] vk (r-15)                ;wait for the drum
[  67] arn -305 t1              ;take the written
[  68] sr -505 t1               ;subtract the read
[  69] gr r15,arn r15           ;store the difference and fetch it again
[  70] hv r15 NZ                ;jump to typeout if not zero
[  71] it (r-3),bs -465         ;this is repeated-
[  72] hv r-5                   ;40 times
[  73] it (r-22),bs 319         ;if track number < 319
[  74] hv r-23                  ;jump back
[  75] hv r42                   ;else jump to typeout of ok
[  76] pm r7,mln r5             ;RAND-1,subroutine for random  numbers
[  77] tk 8,gm r6
[  78] ar r5
[  79] sr r5 L0
[  80] gr r3,hr s1              ;jump back
[  81] gtn p,dln r434           ;constant
[  82] udn (p511),gc -1         ;constant
[  83] sn p206 X t-228          ;constant
[  84] qq
[  85] sy 64,ud r39             ;type CR and count down in 120
```

```
[  86] arn(r-35) D          ;fetch track number
[  87] hs r17               ;jump to decimal typeout
[  88] qq
[  89] arn(r-21) D          ;fetch word number relatively


[  90] sr -504 D            ;find word number
[  91] hs r13               ;jump to decimal typeout
[  92] qq
[  93] sy 64                ;type CR
[  94] arn(r-27),tk 1       ;fetch the written word and-
[  95] sy 1 V LT            ;write-
[  96] sy                   ;it-
[  97] hh r-3 NZ            ;until the rest is only o-es
[  98] sy 64                ;write CR
[  99] arn(r-31),tk 1       ;fetch the read word and-
[ 100] sy 1V LT             ;write it-
[ 101] sy                   ;until the rest is only o-es
[ 102] hh r-3 NZ            ;
[ 103] hvr-32               ;jump to repeat
[ 104] ck 10,tk 30          ;decimal typeout
[ 105] ck -10,pt r9
[ 106] dk r9 X
[ 107] pp 4,mln r9
[ 108] pp p-1,tk 20
[ 109] ar r5 V LZ
[ 110] pt r4 t16
[ 111] gt r , sy
[ 112] bs p510,ud r-2
[ 113] bs p , hh r-6
[ 114] hr s1,qq 16          ;jump back
[ 115] qq9  t-241
[ 116] qq  IZA
[ 117] sy 38 ,sy 34         ;type ok
[ 118] sy ,sy               ;type two spaces
[ 119] sy ,sy               ;type two spaces
[ 120] bt 12 t-1            ;up until 12 times
[ 121] hv 43                ;jump to repeat
[ 122] pa r-2 t12           ;else place 12 as counting number
[ 123] sy 64 ,hv 43         ;write CR and jump to repeat
[ 124] pa r-4 t-1           ;count down to second to the last
e41
```

[Testprogram no. 2, typeout of bit-configuration or bit-pattern.Set the TYPEWRITER to the shortest distance between the lines.
Program is stored from location 41 to 89.The first instruction to be executed is zq LKB.The program will stop after 20 lines have been typed if KB=1.The typeout will be repeated by pressing START-button.]

```
i=41
[  41] zq LKB                    ;stop if KB=1
[  42] vy16 ,hs 50               ;choice of typewriter and jump
[  43] qq 70 t1                  ;constant and counter
[  44] bt 19 t-1                 ;is performed-
[  45] hh r-3                    ;20 times
[  46] zq  LKB                   ;stop if KB=1
[  47] pa r-3 t19                ;set number of words
[  48] pa r-5 t70                ;set start address
[  49] hh r-7                    ;go to repeat
[  50] arn(s1) IPC               ;typeoutprogram.Fetch a word
[  51] sy 64 , sy 60             ;type CR and upper case
[  52] sy 6, tk 1                ;type [ and shift
[  53] sy 57 V LT                ;type I if sign is-
[  54] sy                        ;else type space
[  55] bt 39 t-1                 ;is performed
[  56] hh r-4                    ;40 times
[  57] pa r-2 t39                ;set number of bits
[  58] sy 57 V LPA               ;type I if A-mark
[  59] sy                        ;else type space
[  60] sy 57 V LPB               ;type I if B-mark
[  61] sy                        ;else type space
[  62] sy 7,hr s1                ;type ] and jump back
[  63] qq                        ;empty
[  64] qq
[  65] qq
[  66] qq
[  67] qq
[  68] qq
[  69] qq
[  70] qq s1  IB t1              ;1 word in bitpattern
[  71] qq p3 IC t3
[  72] qq (p7) IKC t7
[  73] qqn (p15) IOC t15
[  74] zqn (p31) IRC t31
[  75] srn (p63) MRC t63
[  76] scn (p127) LRC t127
[  77] nkn (p255) D LRC t255
[  78] ppn (p511) VD LRC t511
[  79] udn (p-1) XVD LRC t-1
[  80] udn (p-1) XVD LRC t-1
[  81] udn (r-2) XVD LRA t-2
[  82] udn (-4) XVD LR t-4
[  83] udn -8 XVD LQ t-8
[  84] ud -16 XVD LT t-16
[  85] zl -32 XVD L t-32
[  86] hh -64 XVD N t-64
[  87] hv -128 XVD t-128
[  88] pc -256 XV t-256
[  89] pa -512 X t-512           ;last word in bitpattern
e41
```

[Testprogram no. 3.Typeout of track no. 0 in bit-pattern. Set the TYPEWRITER to double distance between the lines.
The testprogram is stored in locations 41 to 63,it uses the locations 64 to 103 for storing. The first instruction to be performed is zq LKB.The run can be repeated by pressing the START-button.]

```
i=41
[  41] zq LKB                ;stop if KB=1
[  42] vk 0,1k 64            ;choose and read track no. 0
[  43] vy 16,vk 0            ;choose typewriter and wait for the drum
[  44] sy 62,hs 51           ;black ribbon and jump to typeoutprogram
[  45] qq 64 t1              ;constant and counter
[  46] bt 39 t-1             ;is performed-
[  47] hh r-3                ;40 times
[  48] pa r-3 t64            ;set start address
[  49] pa r-3 t39            ;set number of words
[  50] zq 0,hv r-7           ;stop and go to restart
[  51] arn (s1) IPC          ;typeoutprogram.Fetch a word
[  52] sy 64,sy 60           ;type CR and upper case
[  53] sy 6,tk 1             ;type [ and shift
[  54] sy 57 V LT            ;type I if sign is-
[  55] sy 4                  ;else type =
[  56] bt 39 t-1             ;is performed-
[  57] hh r-4                ;40 times
[  58] pa r-2  t39           ;set number of bits
[  59] sy 57 V LPA           ;type I if A-mark
[  60] sy 4                  ;else type =
[  61] sy 57 V LPB           ;type I if B-mark
[  62] sy 4                  ;else type =
[  63] sy 7 ,hr s1           ;type ] and jump back
[  64] qq                    ;empty
e41
```

[Testprogram no. 4.Test of the fast memory or core-store.The content of M is stored in a location and is read back.If the read is equal to the written, the program will continue with the next location.The test is performed for the locations 60 - 1022.If KB=1 the program will stop and a new content can be placed in M.Position 40 and 41 in a location are not included in the test.By errors <f> is typed on the typewriter every time a content of a location is not hit right.If KA=1 GIER will stop while the number of the location that was hit wrong is placed in the in-register, and the difference between the read and the written is placed in the R-register.By restart or if KA=0 the program will go on testing the next location.The first instruction is zq LKB.]

```
i=41

[41] zq 0  LKB          ;stop if KB=1
[42] gm 43 V            ;store M in location no. 43
[43] qq 0               ;empty
[44] gm 59 M t1         ;store M in running location
[45] arn 43             ;add content of location no. 43
[46] sr (44)            ;subtract running location
[47] hv 52 NZ           ;jump if R≠0
[48] arn 44             ;add content of location no. 44
[49] ca 1022            ;if number of running location is 1022
[50] pa 44 t59          ;set 59 as address in location no. 44
[51] hv 41              ;go to restart
[52] vy 16, sy 54       ;choose typewriter and type f
[53] pi (44)            ;place no. of location in in-register
[54] zq 0 LKA           ;stop if KA=1
[55] bt 40 t-1          ;until 40 times
[56] hv 48              ;jump
[57] pa 55 t40          ;else place 40 as address in location 55
[58] sy 64 , hv 48      ;type CR and jump
[59] qq 0               ;empty

e41
```

[Testprogram no. 5. Test of track no. 0.
The program reads drum track no. 0 to the locations 128-167 in the core store and compares for each location with the correct content of track no. 0 , that is read in to the locations 70-109 after the program. If there is no error in track no. 0 the message ok will be typed. If there is an error in one or more locations, the program will stop after having typed -i- for errors in positions 0-39 and -m- for errors in the mark-positions. The content of the wrong location can be observed in the R-register and the address of the wrong location in the s-register (read modulo 128). In the M-register is -by errors in pos. 0.39 - placed the difference from the subtraction and - by errors in the mark-bits - the correct mark-bits in position 8 and 9.]

i=41

```
[41] zq  LKB
[42] vy 16 , sy 58              ;the typewriter in lower case
[43] zq LKB
[44] vk  0, lk 128              ;read track 0 to loc. 128  and on
[45] vk ,pi                     ;wait for the drum
[46] arn 69 IRC +1              ;correct run cellcontent to R (markbits indic.)
[47] sr 127  +1                 ;subtract next variable location
[48] gi 66 , pp                 ;correct marks to location no. 66
[49] hv 60 NZ                   ;jump. if R ≠ 0
[50] arn (47) IRC               ;read marks indic.
[51] arn 66 IK,                 ;correct marks to R, read marks to p
[52] ncp , hv 63                ;jump if wrong marks
[53] arn 47,nc 167              ;typeout if 40 locations are tested
[54] hh 45                      ;or jump back
[55] sy 64                      ;type CR
[56] sy 38, sy 34               ;type ok
[57] pa 46  +69                 ;replace address number
[58] pa 47  +127                ;replace address number
[59] hv 43
[60] sy 57  X                   ;type -i- ,difference to M
[61] arn (47) , ps(47)          ;wrong cellcontent to R,address to s
[62] zq  ,hv 50
[63] sy 36  X                   ;type -m- ,correct marks to M pos. 8 and 9
[64] arn(47), ps(47)            ;wrong cellcontent to R,address to s
[65] zq , hv 53
[66] qq                         ;location used for storing during run
```

[From location 70 to 109 is now placed the correct content of track no. 0. In this version it is assumed , that the HP-button is blocked by position 0 in the by-register. For a GIER, where the HP-button is blocked by pos. 3 in the by-register, the program can be used, if you before the run change the right half-word instructions in loc. 72 and 103 to vy r79 and vy 80]

```
i=70
[70]  qq
[71]  it 1,pa 10
[72]  gk 1, vy r-497          ;gk 1,vy r79 for HP blocked by pos. 3 in by-reg.
[73]  gi 2  IPC
[74]  gm 3  MPC
[75]  vk 32  IOB t-1
[76]  gr 4  MOB
[77]  tl -39 , pm 24
[78]  gr 7, tln -19
[79]  arn 2, cm -1
[80]  pmfn r  IRB
[81]  pi  66  IZA t-511
[82]  hh  17  NZA
[83]  gp  8 , pp  38
[84]  vk  319 t-1
[85]  sk -24  t-40
[86]  bt  23  t-1
[87]  hv  1 ,  pp  294
[88]  gs  6 , gin  9
[89]  vk  25 , lk -64
[90]  vk  p , lk -73
[91]  vk(5) ,lk -124
[92]  ps  41, ud  5
[93]  ps  s-1,  ar s-125
[94]  bs  s-1,hv r-1
[95]  bs (5)  IOB t-1
[96]  pp  () , hv 21
[97]  sy  64 V NZ
[98]  arn -25, hv -29
[99]  sy  29, sy  60
[100] sy  54, sy 53
[101] sy  33, sy  35
[102] gr -512  MOB
[103] ly 33, vy -496          ;ly 33,vy 80 for HP blocked by pos. 3 in by-reg.
[104] pmn  64 XD IZA
[105] tl  -7 ,ly r1
[106] pi   LZA t508
[107] xr  X  IZB
[108] hv  35  LZB
[109] gr  41  MRC t-1
e41
```

[Testprogram no. 6.Test of the core-store including marks.
The program examines the whole core-store exept for the locations that are occupied by the program itself.
At first zeroes are send to a location and the answer is checked, then ones are send to the same location and the answer is checked.

If stop in location 44 (r1=45): R0-39 ≠ 0, existing ones are wrong.

If stop in location 46 (r2=47): R40-41 ≠ 0, existing ones are wrong.

If stop in location 50 (r1=51): R0-39 ≠ 0, to position(s) containing a one and having a zero on the right side a zero has been send from the store instead of a one.

If stop in location 52 (r1=53): R=40.41 ≠ 1.1 , existing zeroes are wrong.

With LKB=1 the program will stop after having examined the core store from location 58 to 1023 and further on from location 0 to 40.In the indicator, the address of the location that is tested, can be observed.]

```
i=41

[41] zq LKB
[42] grn 57  M +1              ;zero to variable location
[43] arn(42) ,pi (42)
[44] zq NZ                     ;stop if R0-39 ≠ 0
[45] qq  V NC
[46] zq                        ;stop if R40.41 ≠ 0.0
[47] srn 57                    ;constant is subtracted
[48] gr (42) MC                ;ones to variable location
[49] arn 57 , ar (42)          ;constant to R,content of location is added
[50] zq NZ                     ;stop if R0-39 ≠ 0
[51] qq  V LC
[52] zq                        ;stop if R40.41 ≠ 1.1
[53] arn 42 ,nc 40
[54] hv 42                     ;examine the next (variable) location
[55] pa 42 +57                 ;replace the address 57 in location 42
[56] hv 41                     ;restart with location no. 58
[57] qq IB                     ;constant( 1 bit in position 39)

e41
```

[Testprogram no. 7. Topsøes test of the floating point operations: AR F,SR F, MK F and DK F.
The testprogram operates in 6000 cycles.In each cycle two random,floating point numbers,a and b, are generated by means of the subroutines RAND-1 and RAND-2.The four floating point operations are then carried out on the numbers a and b, and the four results are added(as fixed points numbers) to four sum cells.
After 6000 cycles a line is printed on the typewriter,containing the number 6000 and the four sum cells in octal notation.The four sum cells are then cleared and 6000 new cycles are calculated,etc.If KB=1 the program will stop after 6000 cycles, and can be repeated by pressing the NORMAL START-button.

The correct output is (only the numbers):

```
6000  352714077530 20  362427066070 00  660437256575 30  155777777337 44
6000  352714077530 20  362427066070 00  660437256575 30  155777777337 44
etc.
test of:    add.          sub.          mult.          div.
```

Calculation of 6000 cycles takes about 35 sec.The value of 6000 is stored in cell 114 as an integer in pos. 39.It may be changed if required.
If KA is set to 1, output is obtained after each cycle.

The program is stored from cell 41 to 136.he first instruction to be executed is zq LKB.]

i=41

```
[ 41]  zq LKB                    ;stop if KB=1
[ 42]  vy 16,hv r4               ;typewriter output,go to 46
[ 43]  qq 0
[ 44]  qq 0
[ 45]  zq LKB                    ;stop if KB=1
[ 46]  grn r71 [117]             ;sum:=0
[ 47]  grn r71 [118]             ;sum:=0
[ 48]  grn r71 [119]             ;sum:=0
[ 49]  grn r71 [120]             ;sum:=0
[ 50]  hv r19                    ;go to 69
[ 51]  qq 0
[ 52]  qq 0
[ 53]  qq IZA                    ;parameter
[ 54]  gcn s9  t320              ;parameter
[ 55]  pi      t-65              ;from 55 to 66 is placed -
[ 56]  arn r-2,gr r-13           ;a subroutine for printing -
[ 57]  dln r-14,tk 20            ;out the counter in location -
[ 58]  pi  64  NZ t-65           ;115 in decimal notation.
[ 59]  hv  r3   NZ
[ 60]  hv  r2   NTB
[ 61]  sy  16,hv  r2
[ 62]  gt  r ,sy
[ 63]  gm  r-11,pm  r-20
[ 64]  dln  r-11,gr  r-21
[ 65]  hr s2   LZ
[ 66]  pm  r-14,hv  r-9
[ 67]  qq        t256            ;parameter
[ 68]  qq        IB              ;parameter
[ 69]  grn  r-25                 ;zeroes to cell 44
[ 70]  arn  r39,gr  r65          ;mge start-random from 109 to 135
[ 71]  grn  r44                  ;counter:=0
[ 72]  pp  2 pp  p-1             ;p:=2 , p:=p-1
[ 73]  grn  p121,hs  r51         ;cell:=0 , go to RAND-2
[ 74]  qq  20                    ;parameter
```

```
[ 75]  sr  10 D                ;-10
[ 76]  ga  p121,hs  r52        ;set exp. ,go to RAND-1
[ 77]  tk  2    ITA            ;set sign
[ 78]  ck  -6,tl  -6           ;Roo:=0
[ 79]  ac  p121,arn  r-12      ;store , fetch 1
[ 80]  tk  1   LTA             ;if neg. then -2
[ 81]  ac  p121                ;add to cell
[ 82]  bs  p,hh  r-10          ;repeat
[ 83]  pp  4,pp  p-1           ;p:=4 , p:=p-1
[ 84]  arfn  r37,udf  p110     ;fetch b , execute
[ 85]  grf  r38                ;store in 123
[ 86]  arn  r37,ac  p117       ;add to cell
[ 87]  bs  p,hh  r-4           ;repeat
[ 88]  hsn  r5 X LKA t52       ;check output
[ 89]  arn  r-21,ar  r26       ;count
[ 90]  gr  r25,sr  r24         ;- limit
[ 91]  hv  r-19   LT           ;repeat
[ 92]  psn  r43                ;final output
[ 93]  sy  64,arn  r22         ;CAR RET,counter to R-reg.
[ 94]  hs  r-39 X  t59         ;print counter
[ 95]  qq  257  t-144          ;parameters
[ 96]  sy  ,sy                 ;2 SPACES
[ 97]  pp  4,pp  p-1           ;p:=4 , p:=p-1
[ 98]  arn  p117,tl  -10       ;fetch number , shift
[ 99]  ga  r17,sr  r17         ;delete bits 0-9
[100]  tl  3,ca                ;shift , if zero
[101]  sy  16,hv  r2           ;then print zero
[102]  ga  r14,sy (r14)        ;store and print
[103]  bt  13  t-1             ;repeat 13 times
[104]  hv  r-5
[105]  pan  r-2  X  t13        ;reset address in bt-instruction
[106]  sy  ,sy                 ;2 SPACES
[107]  bs  p,hh  r-10          ;repeat
[108]  hr  s1                  ;exit ( to 136)
[109]  bs  p104  V  t-333      ;start random
[110]  dkf  r11                ;/b
[111]  mkf  r11                ;Xa
[112]  srf  r10                ;-a
[113]  arf  r9                 ;+a
[114]  qq  (s)   XV  LT        ;6000 = limit
[115]  qq                      ;counter
[116]  qq                      ;storage of address
[117]  qq                      ;storage of div.
[118]  qq                      ;storage of mult.
[119]  qq                      ;storage of subtr.
[120]  qq                      ;storage of add.
[121]  qq                      ;storage of b
[122]  qq                      ;storage of a
[123]  qq                      ;storage of result
[124]  hs  r4  t44             ;RAND-2
[125]  xr  ,mkn  s1
[126]  mb  511 D
[127]  hr  s2,
[128]  pm  r7,mln  r5          ;RAND-1
[129]  tk  8,gm  r6
[130]  ar  r5
[131]  sr  r3  LO
[132]  gr  r3,hr  s1
[133]  gtn  p,dln  r434
[134]  udn (p511),pc  -1
[135]  qq                      ;age of start random
[136]  hv  45                  ;go to 6000 new cycles
e41
```

[Testprogram no. 8. Test of drum track transfer time.
The time for execution of an LK instruction is tested by the program. If KA is set to 1, the transfer time for all tracks is printed out. If KA=0 there will only be output for tracks with a transfer time different from 20 ms. The output is in binary notation for the transfer times as well as for the track numbers.

After the end of a test run (output <slut> on the typewriter), the test can be repeated by pressing the NORMAL START-button. If the qq instruction in location 57 is changed to a zq instruction, the actual transfer time can after stop in location 57 be observed in the R-register in microseconds with unit in pos. 19.
First instruction to be executed is zq LKB in location 41.]

```
i=41
[41]  zq LKB              ;stop if KB=1
[42]  vy 16,sy 64         ;select typewriter,CR
[43]  sy 58,sy 36         ;lower case,m
[44]  sy 18               ;s
[45]  bt 10 t-1           ;write 10 SPACE
[46]  sy  ,hv 45
[47]  sy 19,sy41          ;t,r
[48]  sy 49,sy 51         ;a,c
[49]  sy34,sy 64          ;k,CR
[50]  bt 320 t-1          ;select -
[51]  vk  -1  V  t1       ;- tracks 0-319
[52]  hv 75               ;go to 75 if track no. =319
[53]  lkn -512            ;read track, zeroes to R
[54]  hk 57               ;if transfer finished then go to 57 -
[55]  ar 74               ;else add 113 microseconds
[56]  hv 54               ;and go to 54
[57]  qq                  ;empty
[58]  nc 20 NKA           ;output for all tracks if KA=1 -
[59]  ps 66 V             ;- else only output for tracks -
[60]  hv 50               ; - with transfer time ≠ 20 ms
[61]  tk 1                ;shift 1 position to the left
[62]  sy 1  V LT          ;print 1 if sign
[63]  sy 38               ;else print o
[64]  bt 9 t-1            ;and repeat 9 times -
[65]  hv 61               ;- from 61
[66]  hv s1
[67]  sy  ,sy             ;2 SPACE
[68]  pa 64 t9            ;reset address in loc. 64
[69]  arn (51) D          ;track no. to R
[70]  hs 61               ;print track no.
[71]  pa 64 t9            ;reset address in loc. 64
[72]  sy  64              ;CR
[73]  hv  50              ;go to new track
[74]  qq  t113            ;number of microseconds
[75]  sy  18              ;s
[76]  sy  35,sy  20       ;l,u
[77]  sy  19,sy  64       ;t,CR
[78]  zq                  ;stop
[79]  pa  45  t10         ;reset loc. 45
[80]  pa  50  t320        ;reset loc. 50
[81]  pa  51t-1           ;reset loc. 51
[82]  hv  41              ;go to start
e41
```

[Testprogram no. 9. Test of the IBM-typewriter for input/output.

When the testprogram is read into GIER, a correct run will be, that every character, that is manually typed on the typewriter, is typed out again 9 times by the program. If upper case is typed, a serie of case shifts will be done.

If KA=0, the characters are tested in lower case, if KA=1, they are tested in upper case.

First instruction to be executed is zq LKB.]

```
i=41
[41]  zq  LKB
[42]  vy 17, ly r3          ;typewriter input/output, read to 45
[43]  sy 60  LKA            ;upper case if KA=1
[44]  sy 58  NKA            ;lower case if KA=0
[45]  qq  0, ca 60          ;if upper case is input-character
[46]  hv r5                 ;then go to caseshift
[47]  pa r1  t9             ;reset address in loc. 48
[48]  bt 9  t-1             ;9 times -
[49]  sy  (r-4),hv r-1      ;- print the read character
[50]  sy 64,hh 42           ;type CR, go to new character
[51]  sy 60, sy 58          ;case shift
[52]  sy 60, sy 58
[53]  sy 60, sy 58
[54]  sy 60, sy 58
[55]  sy 60,  LKA           ;end in upper case if KA=1
[56]  hh 42                 ;go to new character
e41
```

[Testprogram t10 (ferrittest)

The program test reading and writing in cells 512-1023 in the fast memory. If a cell in this interval is mistaken for another cell in the same interval there will be no errorreaction , because the program use the same bitpattern in all cells during one testcycle.The program is stored from cell 41 to 131.

INPUT OF PROGRAM

The program is published in a condensed version (a B-tape) to be read into the machine by the basic inputprogram on channel 0,cells 34-39.After the input a sumcheck of the program is performed.If this check is ok. the program writes <CR,t 10 > on the typewriter with black ribbon, otherwise the programname will be typed in red,and the program stops in cell 49.

TESTDIGITS

After reading in and sumcheck the program will stop in cell 51 if KB=1.During this stop the registers R and In contains zeroes.If no bits are inserted the program itself will generate the testdigits for the run.If bits are inserted in R or In these bits will serve as testdigits during the run.In In only RA and RB gives sense.

BUILD IN TESTDIGITS

When the build in testdigits are used , 42 consecutive bits is taken from a group of bits containing:

1,1,1,1,....,1,0,0,0,0,....,0,1,1,1,1,....,1

42 bits 42 bits 41 bits

This means that each cell is automatically tested with 84 different (but known) bitpatterns.

OPERATORS TESTDIGITS

If the operator choose the testdigits each cell will be tested 81 times with the choosen bitcombination.The operator must insert the bits in R0-39 and the marks in In[RA,RB].

ERROR REACTIONS

During the running of the program , the normal condition of KA and KB is KA=KB=0.In this case there will be no error-output before one complete run is terminated.A complete run means writing and reading in cells 512-1023 of either the 84 generated bitpatterns or 81 times with the operators bits.

NO ERROR: If no error is registrated the run will terminate with the typed message <CR,ok> and a hoot at 1000 c/s in 5 seconds.

ERROR: If an error is registrated the run will terminate with the typed message <CR,-> and the frequency of the hoot will alternate between 950 c/s and 1100 c/s.

KA=1 will cause the program to stop when an error is detected in the bits 0-39 or the marks in the cell in question.There is two possible stoppoints in the program:

1) Stop with r1=124 [000111100] indicates error in bits 0-39. The testbits used will be found in M.Bits in R set to 1 are errors.Depressing the START-button two times while the STOP-button is depressed will transfer the adress of the cell to p and the cellcontent to R0-39.Depressing START will continue the test.
2) Stop with r1=124 [0001111100] indicates error in the marks. The testmarks is stored in R8-9 and the marks of the cell in In[RA,RB].The adress of the cell is transferred to p. Depressing START will continue the test.

Setting KB=1 when the machine is stopped in an error , and depressing START , will cause cycling in a loop as long as KB=1.In this loop th content of M0-39 and In[RA,RB] is stored in the cell , and the content of the cell is read to R0-39 and In[QA,QB].

]

```
[Test of the fast memory,cell 512-1023]
i=  41
[  41] vy   16          ,psn  132
[  42] ps   s-1         ,ar   s
[  43] bs   s-41        ,hv   r-1 [42]   ;form checksum of the program
[  44] sy   62          ,tk   1
[  45] sy   29            NZ             ;if errorsum then redshift
[  46] sy   64          ,sy   19
[  47] sy               ,sy   1
[  48] sy   16          ,sy   62         ;outtext(outcr;«t 10»,blackshift)
[  49] zq                 NZ             ;if errorsum then stop
[  50] pin                               ;R:=In:=0
[  51] zq                 LKB            ;if KB then stop for manual testdigits
[  52] gr   r57 [109]     M              ;controlcell 40,41 := 0,0
[  53] pa   r35 [88]        t1           ;manualdigits:=true
[  54] gi   r2 [56]     V LZ             ;if R=0 then c56adr:=In / goto c56
[  55] pa   r18 [73]    ,hv   r36 [91]   ;error:=false / goto operators testdigits
[  56] nc               ,hv   r-1 [55]   ;if In≠0 then goto c55
[  57] pa   r31 [88]    ,pa   r16 [73]   ;manualdigits:=false / error:=false
[  58] pm    D                           ;M:=0
[  59] pi               ,hs   101        ;In:=0 / test
[  60] pi   1           ,hs   101        ;In:=1 / test
[  61] pi   3           ,hs   101        ;In:=3 / test
[  62] pm   r37 [99]    ,hs   101        ;M 0-39 := all 1 / test
[  63] xrn              ,cl   -1         ;R:=M×2↑(-1)
[  64] hh   r-2 [62]    X  NZ           ;if M≠0 then begin M:=R;goto c62 end
[  65] pm   r34 [99]    ,pi   2          ;M 0-39 := all 1 / In:=2
[  66] hs   101                          ;test
[  67] pi               ,hs   101        ;In:=0 / test
[  68] xr               ,tk   1          ;R:=M×2
[  69] pm   r30 [99]    ,cm   r31 [100]  ;if R≠-1 then begin M:=R;goto c71 end
[  70] hh   r-3 [67]    X                ;else goto c67
[  71] gr   r38 [109]   XV MRC           ;controlcell 40,41 := 1,1 / M:=R
[  72] hs   101                          ;test
[  73] ps               ,sy   64         ;s:=error / outcr
[  74] can  s           ,hv   r2 [76]    ;if -,error then goto c76
[  75] sy   32          ,hv   r2 [77]    ;else writetext(«<-») / goto c77
[  76] sy   38          ,sy   34         ;writetext(«<ok»)
[  77] pa   r9 [86]       t9
[  78] pa   r5 [83]       t511           ;initialize hootduration
[  79] pa   r2 [81]       t7             ;initialize frequency-cycle
[  80] ar   s128        D                ;Radr := (if error then 8 else 0)+128+Radr
[  81] bt   7             t-1            ;for i:=7 step -1 until 1 do
[  82] hv   r-2 [80]                     ;goto c80
[  83] bt   511           t-1            ;for j:=511 step -1 until 1 do
[  84] hv   r-5 [79]                     ;goto c79
[  85] ns   s           ,ps   s          ;s:=-s
[  86] bt   9             t-1            ;for k:=9 step -1 until 1 do
[  87] hv   r-9 [78]                     ;goto c78
[  88] bs                                ;if manualdigits then
[  89] hv   r-38 [51]   X                ;goto newtest(Operatorsdigits)
[  90] hv   r-40 [50]                    ;else goto newtest(buildindigits)
[  91] pm   r8 [99]     ,cm   r9 [100]   ;Operatorsdigits:
[  92] hv   r2 [94]                      ;if R≠-1 then goto c94
[  93] gr   r16 [109]     MRC            ;controllcell 0-39 := R ;40;41:=RA;RB
[  94] pa   r2 [96]     X     t80        ;M:=R / for i:=1 step 1 until 81 do
[  95] hs   101                          ;test
```

```
[  96] bt                          t-1
[  97] hv    r-2 [95]
[  98] hv    r-25 [73]                      ;test finished
[  99] udn (p-1)          XVDLRC t-1        ;constant 0-39 := all 1
[ 100] qq    -512                           ;procedure test;begin
[ 101] pa    r3 [104]                t511  ;entry to test
[ 102] pa    r1 [103]                      ;for j:=1023 step -1 until 512 do
[ 103] gm                          MRC t-1 ;cell j 0-39:= M cellj 40-41 :=RA,RB
[ 104] bt    511                     t-1
[ 105] hv    r-2 [103]
[ 106] pa    r12 [118]               t511  ;for i:=1023 step -1 until 512 do begin
[ 107] pa    r7 [114]
[ 108] grn   r1 [109]     XV               ;controlcell 0-39 := 0 / R:=testdigits
[ 109] qq
[ 110] sc    r-1 [109]    X                ;controlcell:=-testdigits / M:=testdigits
[ 111] gi    r9 [120]                      ;c120adr:=In
[ 112] pi                         ,arn  r-3 [109]  ;In:=0 / R:=-testdigits
[ 113] srn   r-4 [109]              LC             ;if R=-1 then R:=-(-testdigits)
[ 114] ar                           IRC t-1        ;R:=R+cell[i] / In[RA,RB]:=cell i 40,41
[ 115] hv    r6 [121]               NZ             ;if R≠0 then goto errorincell
[ 116] gi    r1 [117]    ,arn  r4 [120]  ;c117adr := In / R:= old In
[ 117] nc                ,hh   r6 [123]  ;if newmarks≠oldmarks then goto errorinmarks
[ 118] bt                       t-1      ;one cell tested
[ 119] hv    r-7 [112]
[ 120] pi                ,hr   s1,       ;In:=Original marks / end of procedure test;
[ 121] zq                   LKA          ;errorincell: if errorincell∧KA then stop
[ 122] pp  (r-8)[114]    ,arn  p         ;p:=adress of wrong cell / R:=wrong cell
[ 123] hv    r2 [125]    ,pp  (r-9)[114] ;goto seterror / errorinmarks: p:=adress of wro
[ 124] zq                   LKA          ;if error in marks ∧KA then stop
[ 125] pa    r-52 [73]          t8       ;seterror: error:=true
[ 126] pi  (r-6)[120]    ,hv   r3 [129]  ;In:=original marks
[ 127] gm    p              MRC          ;testdigits to cell
[ 128] arn   p              IQC          ;cell to R
[ 129] hv    r-2 [127]      LKB          ;if KB then cycle in 127-128
[ 130] pi  (r-10)[120]   ,hv   r-12 [118] ;In:=Original marks / continue the test
[ 131] gmn   -319        X  LP  t488      ;checksum of program
s
```

[Testprogram t11 (ferrittest)

The program test reading and writing in cells 0-511 in the fast memory. If a cell in this interval is mistaken for another cell in the same interval there will be no errorreaction , because the program use the same bitpattern in all cells during one testcycle.The program is stored from cell 513 to 603 .

INPUT OF PROGRAM

The program is published in a condensed version (a B-tape) to be read into the machine by the basic inputprogram on channel 0,cells 34-39.After the input a sumcheck of the program is performed.If this check is ok. the program writes <CR,t 10 > on the typewriter with black ribbon, otherwise the programname will be typed in red,and the program stops in cell 521.

TESTDIGITS

After reading in and sumcheck the program will stop in cell 523 if KB=1.During this stop the registers R and In contains zeroes.If no bits are inserted the program itself will generate the testdigits for the run.If bits are inserted in R or In these bits will serve as testdigits during the run.In In only RA and RB gives sense.

BUILD IN TESTDIGITS

When the build in testdigits are used , 42 consecutive bits is taken from a group of bits containing:

1,1,1,1,....,1,0,0,0,0,....,0,1,1,1,1,....,1

42 bits 42 bits 41 bits

This means that each cell is automatically tested with 84 different (but known) bitpatterns.

OPERATORS TESTDIGITS

If the operator choose the testdigits each cell will be tested 81 times with the choosen bitcombination.The operator must insert the bits in R0-39 and the marks in In[RA,RB].

ERROR REACTIONS

During the running of the program , the normal condition of KA and KB is KA=KB=0.In this case there will be no error-output before one complete run is terminated.A complete run means writing and reading in cells 0-511 of either the 84 generated bitpatterns or 81 times with the operators bits.

NO ERROR: If no error is registrated the run will terminate with the typed message <CR,ok> and a hoot at 1000 c/s in 5 seconds.

ERROR: If an error is registrated the run will terminate with the typed message <CR,-> and the frequency of the hoot will alternate between 950 c/s and 1100 c/s.

KA=1 will cause the program to stop when an error is detected in the bits 0-39 or the marks in the cell in question.There is two possible stoppoints in the program:

1) Stop with r1=594[1001010010] indicates error in bits 0-39. The testbits used will be found in M.Bits in R set to 1 are errors.Depressing the START-button two times while the STOP-button is depressed will transfer the adress of the cell to p and the cellcontent to R0-39.Depressing START will continu the test.
2) Stop with r1=597[1001010101] indicates error in the marks. The testmarks is stored in R8-9 and the marks of the cell in In[RA,RB].The adress of the cell is transferred to p. Depressing START will continue the test.

Setting KB=1 when the machine is stopped in an error , and depressing START , will cause cycling in a loop as long as KB=1.In this loop th content of M0-39 and In[RA,RB] is stored in the cell , and the content of the cell is read to R0-39 and In[QA,QB].

]

```
[Test of the fast memory,cells 0-511]
i= -511
[-511] vy   16          ,psn  -420
[-510] ps   s-1         ,ar   s
[-509] bs   s511        ,hv   r-1 [-510]   ;form checksum of the program
[-508] sy   62          ,tk   1
[-507] sy   29            NZ               ;if errorsum then redshift
[-506] sy   64          ,sy   19
[-505] sy               ,sy   1
[-504] sy   1           ,sy   62           ;typetext(CR,<<t 11>>,blackshift)
[-503] zq                 NZ               ;if errorsum then STOP
[-502] pin                                 ;R:=In:=0
[-501] zq                 LKB              ;if KB=1 then STOP for manual testdigits
[-500] gr   r57 [-443]    M                ;controlcell 40,41:= 0,0
[-499] pa   r35 [-464]      t1             ;manualdigits:=true
[-498] gi   r2 [-496]   V  LZ              ;if R=0 then c528adr:=In / goto c528
[-497] pa   r18 [-479]  ,hv   r36 [-461]   ;error:=false / goto operators testdigits
[-496] nc               ,hv   r-1 [-497]   ;if In≠0 then goto c527
[-495] pa   r31 [-464]  ,pa   r16 [-479]   ;manualdigits:= false / error:= false
[-494] pm    D                             ;M:= 0
[-493] pi               ,hs   -451         ;In:= 0 / test
[-492] pi   1           ,hs   -451         ;In:= 1 / test
[-491] pi   3           ,hs   -451         ;In:= 3 / test
[-490] pm   r37 [-453]  ,hs   -451         ;M0-39:= all 1 / test
[-489] xrn              ,cl   -1           ;R:= Mx2↑(-1)
[-488] hh   r-2 [-490]  X  NZ              ;if M≠0 then begin M:=R;goto c534 end
[-487] pm   r34 [-453]  ,pi   2            ;M0-39:= all 1 / In:=2
[-486] hs   -451                           ;test
[-485] pi               ,hs   -451         ;In:= 0 / test
[-484] xr               ,tk   1            ;R:= Mx2
[-483] pm   r30 [-453]  ,cm   r31 [-452]   ;if R≠-1 then begin M:=R;goto c543 end
[-482] hh   r-3 [-485]  X                  ;else goto c539
[-481] gr   r38 [-443]  XV MRC             ;controlcell 40,41:= 1,1 / M:=R
[-480] hs   -451                           ;test
[-479] ps               ,sy   64           ;s:= error / typetext(CR)
[-478] can  s           ,hv   r2 [-476]    ;if -,error then goto c548
[-477] sy   32          ,hv   r2 [-475]    ;else typetext(<<->>) / goto c549
[-476] sy   38          ,sy   34           ;typetext(<<ok>>)
[-475] pa   r9 [-466]       t9
[-474] pa   r5 [-469]       t511           ;initialize hootduration
[-473] pa   r2 [-471]       t7             ;initialize frequencycycle
[-472] ar   s128        D                  ;Radr:= Radr+128+(if error then 8 else 1)
[-471] bt   7               t-1            ;for i:= 7 step -1 until 1 do
[-470] hv   r-2 [-472]                     ;goto c552
[-469] bt   511             t-1            ;for j:= 511 step -1 until 1 do
[-468] hv   r-5 [-473]                     ;goto c551
[-467] ns   s           ,ps   s            ;s:= -s
[-466] bt   9               t-1            ;for k:= 9 step -1 until 1 do
[-465] hv   r-9 [-474]                     ;goto c550
[-464] bs                                  ;if manualdigits then
[-463] hv   r-38 [-501] X                  ;newtest(operators testdigits)
[-462] hv   r-40 [-502]                    ;else newtest(buildin testdigits)
[-461] pm   r8 [-453]   ,cm   r9 [-452]    ;operators testdigits:
[-460] hv   r2 [-458]                      ;if R≠ -1 then goto c566
[-459] gr   r16 [-443]    MRC              ;controlcell0-39:= R , 40-41:=RA-RB
[-458] pa   r2 [-456]   X     t80          ;M:=R / for i:= 1 step 1 until 81 do
[-457] hs   -451                           ;test
[-456] bt                   t-1
[-455] hv   r-2 [-457]
[-454] hv   r-25 [-479]                    ;test finished
[-453] udn (p-1)        XVDLRC t-1         ;constant0-39= all 1
```

```
[-452] qq   -512                                ;procedure test;begin
[-451] pa   r3 [-448]          t511             ;entry to test
[-450] pa   r1 [-449]          t-512            ;for j:= 511 step -1 until 0 do
[-449] gm                  MRC t-1              ;cellj0-39:= M, 40-41:=RA,RB
[-448] bt   511                t-1
[-447] hv   r-2 [-449]
[-446] pa   r12 [-434]         t511             ;for i:= 511 step -1 until 0 do begin
[-445] pa   r7 [-438]          t-512
[-444] grn  r1 [-443]      XV                   ;controlcell0-39:= 0 / R:=testdigits
[-443] qq
[-442] sc   r-1 [-443]     X                    ;controlcell:= -testdogits / M:= testdigits
[-441] gi   r9 [-432]                           ;c592adr:= In
[-440] pi                  ,arn  r-3 [-443]     ;In:= 0 / R:=-testdigits
[-439] srn  r-4 [-443]       LC                 ;if R= -1 then R:=-(-testdigits)
[-438] ar                    IRC t-1            ;R:=R+cell i / In[RA,RB]:= cell i 40,41
[-437] hv   r6 [-431]        NZ                 ;if R≠ 0 then goto error in cell
[-436] gi   r1 [-435]      ,arn  r4 [-432]      ;c589adr:= In / Radr:= old In
[-435] nc                  ,hh   r6 [-429]      ;if newmarks≠ old marks then goto error in marks
[-434] bt                        t-1            ;one cell tested
[-433] hv   r-7 [-440]
[-432] pi                  ,hr   s1             ;In:= original marks / end of procedure test;
[-431] zq                    LKA                ;error in cell: if error∧KA=1 then STOP
[-430] pp  (r-8)[-438]     ,arn  p              ;p:=adress of error / R:=errorcell
[-429] hv   r2 [-427]      ,pp  (r-9)[-438]     ;goto set error / error in marks: p:=erroradress
[-428] zq                    LKA                ;if error∧KA=1 then STOP
[-427] pa   r-52 [-479]          t8             ;set error: error:=true
[-426] pi  (r-6)[-432]     ,hv   r3 [-423]      ;In:= original marks
[-425] gm   p                MRC                ;send testdigits to cell
[-424] arn  p                IQC                ;read cell to R
[-423] hv   r-2 [-425]       LKB                ;if KB=1 then cycle in cells 599-600
[-422] pi  (r-10)[-432]    ,hv   r-12 [-434]    ;In:= original marks / continue with next cell
[-421] gmn  -228           X  LP  t-296         ;checksum of the program
s
```

Before start KA, KB are set to 0, 1. The test is supplied with a number of loops, so if any error occurs, KA is set to one, and the program will run in the loop, where the error appeared. All stops indicate errors. Cellnumber is found by subtraction of one from the content of r_1.

```
41   zq          LKB
42   grn  3oo
43   pi   -1
44   gi   3oo
45   arn  3oo
46   sr   -1  D
47   zq          NZ
48   hv   43     LKA   loop A, test of flip-flops for "ones"
49   pi
5o   gi   3oo    MOC
51   arn  3oo
52   zq          NZ
53   hv   49     LKA   loop B, test of flip-flops for "zeroes"
54   pi   -1           t-2 inhibit changing all except RB
55   zq          NRB
56   pi   -2           t-3 inhibit changing all except RA
57   zq          NRA
58   hv   6o     LRC  }
59   zq                }
6o   zq          LOA   }
61   zq          LOB   }
62   zq          LTA   }
63   zq          LTB   } proving "L" indicator orders
64   zq          LPA   }
65   zq          LPB   }
66   zq          LQA   }
67   zq          LQB  }
68   gi   3oo ,arn  3oo
69   sr   3   D
7o   zq          NZ
```

```
71   qq          MOC   R40-41 are set to zero
72   ppn         IRB   dummy order , RB: = 0
73   ppn         IRA   dummy order , RA: = 0
74   zq          LRB
75   zq          LRA
76   qq          M     R40-41 are set to one
77   ck   1o     IRB   dummy order, RB: = 1
78   ck   1o     IRA   dummy order, RA: = 1
79   zq          NRA
8o   zq          NRB
81   hv   54     LKA   loop C, test of RA, RB
82   pi   -4           t-5 inhibit changing all except QB
83   zq          NQB
84   pi   -8           t-9 inhibit changing all except QA
85   zq          NQA
86   hv   88     LQC  }
87   zq               }
88   zq          LOA  }
89   zq          LOB  }
9o   zq          LTA  } proving "L and N" indicator orders
91   zq          LTB  }
92   zq          LPA  }
93   zq          LPB  }
94   zq          NRA  }
95   zq          NRB  }
96   gi       ,arn    3oo
97   sr   15  D
98   zq          NZ
99   gmn  4oo    IQB   dummy order, QB: = 0
1oo  gmn  4oo    IQA   dummy order, QA: = 0
1o1  zq          LQB
1o2  zq          LQA
1o3  qq          M
1o4  tk   2      IQB   dummy order, QB: = 1
1o5  tk   2      IQA   dummy order, QA: = 1
1o6  zq          NQB
1o7  zq          NQA
1o8  hv   82     LKA   loop D, test of QA, QB
```

```
1o9   pi   -16          t-17 inhibit changing all except PB
11o   zq         NPB
111   pi   -32          t-33 inhibit changing all except PA
112   zq         NPA
113   hv   115  LPC
114   zq
115   zq         LOA
116   zq         LOB
117   zq         LTA    proving "L" and "N" indicator orders
118   zq         LTB
119   zq         NQA
12c   zq         NQB
121   zq         NRA
122   zq         NRB
123   gi   3oo
124   arn  3oo
125   sr   63 D
126   zq         NZ
127   gmn  4oo   IPB    dummy order, PB: = 0
128   gmn  4oo   IPA    dummy order, PA: = 0
129   zq         LPB
13o   zq         LPA
131   qq         M
132   tl   4     IPB    dummy order, PB: = 1
133   tl   4     IPA    dummy order, PA: = 1
134   zq         NPB
135   zq         NPA
136   hv   1o9   LKA    loop E, test of PA and PB
137   pi   -64          t-65 inhibit changing all except TB
138   zq         NTB
139   pi   -128         t-129 inhibit changing all except TA
14o   zq         NTA
141   hv   143  LTC
142   zq
```

```
143   zq          LOA  \
144   zq          LOB   |
145   zq          NTA   |
146   zq          NTB   |
147   zq          NPA   | proving "L" and "N" indicator orders
148   zq          NPB   |
149   zq          NQA   |
150   zq          NQB   |
151   zq          NRA  /
152   zq          NRB
153   gi   300
154   arn  300
155   sr   255    D
156   zq          NZ
157   qqn         ITB   TB: = 0
158   qqn         ITA   TA: = 0
159   zq          LTB
160   zq          LTA
161   arn  -1     D ITB   TB: = 1
162   arn  -1     D ITA   TA: = 1
163   zq          NTB
164   zq          NTA
165   hv   137    LKA   loop F, test of TA and TB
166   pi   -256         t-257 inhibit changing all except OB
167   zq          NOB
168   zq          NZB
169   pi   -512         t-511 inhibit changing all except OA
170   zq          NOA
171   zq          NZA
172   hv   174    LOC  \
173   zq                |
174   hv   176    LZC   |
175   zq                |
176   zq          NTA   |
177   zq          NTB   | proving "L" and "N" indicator orders
178   zq          NPA   |
179   zq          NPB   |
180   zq          NQA   |
181   zq          NQB   |
182   zq          NRA   |
183   zq          NRB  /
```

```
184   gi   3oo
185   arn  3oo
186   sr   -1    D
187   zq             NZ
188   arn            IOB   dummy order, OB: = 0
189   qqn            IOA   dummy order, OA: = 0
19o   zq             LOB
191   zq             LOA
192   arn  -1    D
193   ar   -512  D IOB                  OB: = 1
194   ar   -512  D IOA                  OA: = 1
195   zq             NOB
196   zq             NOA
197   hv   166       LKA   loop G, test of OA, OB
198   pi   -1
199   pi                   t-1
2oo   hv   2o2       LZC  \
2o1   zq                   |
2o2   hv   2o4       LOC  |
2o3   zq                   |
2o4   hv   2o6       LTC  |
2o5   zq                   } proving "L" indicator orders
2o6   hv   2o8       LPC  |
2o7   zq                   |
2o8   hv   21o       LQC  |
2o9   zq                   |
21o   hv   212       LRC  /
211   zq
212   arn  211       IZB  \
213   ar   211       IZA  |
214   zq             LZB  |
215   zq             LZA  |
216   pmn            IZC  |
217   zq             NZB  |
218   zq             NZA  } proving Z combinations
219   pin                 |
22o   qq             IZB  |
221   ar   212            |
222   zq             NZB  |
223   pin                 |
224   qq             IZA  |
225   ar   213            |
226   zq             NZA  /
```

```
227   pi
228   hv   198     LKA   loop H, test of Z
229   hv   41
```

[Tromletest 4. This program writes random numbers on previous selected drum tracks, reads and compares the contents word by word incl. flag bits. Calculations may be performed simultaneously with the drum test by means of an adder test.

After the message ,klar, the typewriter is ready for input of 3 arbitrary numbers (each terminated by CR) intended for the production of the random test numbers.

The program has several adjustment variations. The p-register contains the track no. to be tested. The five last bits of the in-register have the following meaning:

KB=0 : Adder test simultaneously with the drum test.
KB=1 : Either adder test or drum test depending on when KB is set equal to 1.
KA=0 : No stop if adder error.
KA=1 : Stop if adder error.
The typewriter writes ,1, if adder test is ok, and ,3, if adder error.
RB=0 : No counting in the p-register. Test of the same track.
RB=1 : Counting in the p-register. Track no. is increased with 1.
RA=0 : Output of number of errors per track.
RA=1 : Output of number of errors totally only.
QB=0 : Output of the read and written bit pattern, track no. and word no. if error.
QB=1 : No output of the read and written bit pattern, track no. and word no.

By entry a normal adjustment is set intended for a routine test of tracks 1-319 and output only of the number of errors totally (cf. the instructions in cell 42). By installations with Three Drum Cabinet the instruction in cell 51 should be changed to: arn 959 D.

The only way to change the normal adjustment is to change the contents of the in- and p-registers manually and jump to cell 43. The program starts with the instruction ,zq 0 LKB, and stops only if the operator interferes.]

```
i= 41
[  41] zq            LKB        [ Stop if KB=1]
[  42] pi    7     ,pp           [ Normal adjustment. Start on track 1]
[  43] gp   53     ,hs   89      [ Jump to input of 3 numbers]
[  44] pt   61           t45     [ Initialising return jump to drum test]
[  45] hv  257     ,grn  76      [ Start adder test, clear error counter tot.]
[  46] pp   p1       LRB         [ Add 1 to track no.]
[  47] hs  187                   [ Store inputed numbers]
[  48] hv  155     ,hv   191     [ Fill track, store numbers]
[  49] grn  77     ,hs   187     [ Clear error counter per track, store numbers]
[  50] hv  164     ,hv   199     [ Read track, transfer numbers]
[  51] arn 319     D             [ If more tracks]
[  52] ncp         ,hv   46      [ then jump to new test]
[  53] pp          ,sy   64      [ else reset start track, write CR]
[  54] arn  76     ,hs   203     [ Jump to output of number of errors totally]
[  55] sy   64     ,hh   45      [ Write CR, jump to new test]
[  56] pt   61           V163    [ Entry interrupt drum test]
[  57] pt   61           t168    [ Entry interrupt drum test]
[  58] pm  195     ,arn  196     [ Reset R and M before]
[  59] gs   61     ,hv   290     [ jump to adder test]
[  60] gm  195     ,gr   196     [ Entry interrupt adder test]
[  61] ps          ,hh           [ Jump to drum test]
[  62] qq
```

```
[  63] qq                              [ Working cells]
[  64] qq
[  65] qq
[  66] qq
[  67] qq
[  68] qq
[  69] qq
[  70] qq
[  71] 0/1023/1023/1023                [ Constant]
[  72]     1                           [ Constant]
[  73] qq
[  74] qq
[  75] qq
[  76] qq
[  77] qq


                                       [ Subroutine, produce random test number]
[  78] pm   68                         [ From the numbers in cells 66, 67, and 68]
[  79] arn  68    X                    [ 3 numbers are produced to be placed in the]
[  80] mk   66                         [ same cells. The contents of R is on exit]
[  81] sc   67                         [ depending on the contents of these cells]
[  82] dl   68    X                    [ and used as a test number. Possibly]
[  83] ac   66      IOA                [ overflow is indicated in respective OA]
[  84] ml   67                         [ and OB for later check of flag bits.]
[  85] ac   68      IOB
[  86] hr   s1


                                       [ Subroutine, input of 3 numbers]
[  87]   0                             [ Working cell]
[  88]  10                             [ Constant]
[  89] pa   106          t62           [ Reset start cell]
[  90] pa   107          t2            [ Reset counting]
[  91] vy   17    ,sy    64            [ Select typewriter as I/O, write CR]
[  92] sy   34    ,sy    35            [ Write k, write l]
[  93] sy   49    ,sy    41            [ write a, write r]
[  94] sy   64                         [ Write CR]
[  95] pt   102          t106          [ Set jump address for positive number]
[  96] xrn        ,lyn   87            [ Clear M, input of 1 character]
[  97] nc   32    ,hv    100           [ If character ≠32 then jump]
[  98] pt   102          t105          [ else set jump address for negative number]
[  99] hh   96                         [ Jump to input of next character]
[ 100] ca   16    ,hvn   103           [ If character=0 then clear R and jump]
[ 101] ca   64                         [ If character=64]
[ 102] xr         ,hv                  [ then exchange M and R and jump to storing]
[ 103] tk   -30   ,ml    88            [ else produce a decimal number in M]
[ 104] hh   96                         [ Jump to input of next character]
[ 105] mt   -1    D                    [ If negative number then change sign]
[ 106] gr   62           t1            [ Store the decimal number in working cell]
[ 107] bt   2            t-1           [ Counter for input of 3 numbers]
[ 108] hv   95                         [ Jump within 3 numbers]
[ 109] hr   s1                         [ else exit]


                                       [ Subroutine, output of bit pattern from R]
[ 110] pa   121          t3            [ Prepare output in 4 groups]
[ 111] tl   -10                        [ Prepare output of 1 bit at a time]
[ 112] pa   118          t9            [ Prepare 10 positions in each group]
[ 113] mb   71                         [ Clear R pos. 0-9]
```

```
[ 114] tl   1                       [ Produce 1 bit]
[ 115] ca                           [ If it is 0]
[ 116] sy   16    V                 [ then write 0]
[ 117] sy   1                       [ else write 1]
[ 118] bt   9           t-1         [ Counter for output of 10 pos.]
[ 119] hv   113                     [ Jump back within a group]
[ 120] sy                           [ else write SP]
[ 121] bt   3           t-1         [ Counter for output of 4 groups]
[ 122] hv   112                     [ Jump back within 4 groups]
[ 123] sy   1     V  LPA            [ else write ,1, or ,0, according to PA]
[ 124] sy   16                      [ which corresponds to bit 40]
[ 125] sy   1     V  LPB            [ Write ,1, or ,0, according to PB]
[ 126] sy   16                      [ which corresponds to bit 41]
[ 127] hr   s1                      [ Exit]


                                    [ Subroutine, output of track no. and word no.]
[ 128] sy         ,sy               [ Write SP, write SP]
[ 129] sy   34    ,sy   49          [ Write k, write a]
[ 130] sy   37    ,sy   49          [ write n, write a]
[ 131] sy   35    ,sy               [ write l, write SP]
[ 132] arn  p     D                 [ Prepare output]
[ 133] ck         ,tl   -30         [ of track no.]
[ 134] hs   203                     [ Jump to output of track no.]
[ 135] sy         ,sy               [ Write SP, write SP]
[ 136] sy   38    ,sy   41          [ Write o, write r]
[ 137] sy   52    ,sy   37          [ write d, write n]
[ 138] sy   41    ,sy   59          [ write r, write .]
[ 139] srn  314   D                 [ Prepare output]
[ 140] ar   170   ,tl   -30         [ of word no.]
[ 141] hs   203                     [ Jump to output of word no.]
[ 142] hr   s1                      [ Jump back to error output]


                                    [ Subroutine, error output]
[ 143] arn  72    ,ac   76          [ Add 1 to number of errors totally]
[ 144] ac   77                      [ Add 1 to number of errors per track]
[ 145] hv   180     LQB             [ If QB=1 then jump back without output]
[ 146] arn  69      IPC             [ else prepare output of test number]
[ 147] vy   16    ,sy   64          [ Select typewriter, write CR]
[ 148] sy   64                      [ Write CR]
[ 149] hs   110                     [ Jump to output of bit pattern]
[ 150] sy   64                      [ Write CR]
[ 151] arn (170)    IPC             [ Prepare output of read test number]
[ 152] hs   110                     [ Jump to output of bit pattern]
[ 153] hs   128                     [ Jump to output of track no. and word no.]
[ 154] hv   180                     [ Jump back to next test]


                                    [ Fill drum track]
[ 155] pa   158         t215        [ Reset start address]
[ 156] pa   159         t39         [ Reset counting]
[ 157] hs   78                      [ Jump to produce random test number]
[ 158] gr   215     MOC t1          [ Store result incl. flag bits]
[ 159] bt   39          t-1         [ Counter for 40 words]
[ 160] hv   157                     [ Jump back within a track]
[ 161] vk   p     ,sk   216         [ else select actual track no., write on track]
[ 162] hv   56      NKB             [ Interrupt if KB=0]
[ 163] vk   p     ,hh   48          [ Exit]
```

```
                                          [ Read track and check]
[ 164] pa   180          t39              [ Reset counting]
[ 165] pa   170          t313             [ Reset start address]
[ 166] vk   p     ,lk    314              [ Select actual track no., read track]
[ 167] hv   57      NKB                   [ Interrupt if KB=0]
[ 168] vk   p     ,hs    78               [ Jump to produce random test number]
[ 169] gr   69      MOC                   [ Store test number incl. flag bits]
[ 170] sr   313          t1               [ Subtract read number]
[ 171] hv   143     NZ                    [ Jump to error output if R pos. 0-39≠0]
[ 172] hv   175     LOB
[ 173] hv   143     LB                    [ Jump to error output if OB=0∧b=1]
[ 174] hv   176
[ 175] hv   143     NB                    [ Jump to error output if OB=1∧b=0]
[ 176] hv   179     LOA
[ 177] hv   143     LA                    [ Jump to error output if OA=0∧a=1]
[ 178] hv   180
[ 179] hv   143     NA                    [ Jump to error output if OA=1∧a=0]
[ 180] bt   39           t-1              [ Counter for 40 words]
[ 181] hv   168                           [ Jump back within a track]
[ 182] hh   50      LRA                   [ else if RA=1 then return jump]
[ 183] vy   16    ,sy    64               [ else select typewriter, write CR]
[ 184] arn   77                           [ Prepare output of number of errors per track]
[ 185] hs   203                           [ Jump to output of number of errors]
[ 186] sy   64    ,hh    50               [ Write CR, exit]

                                          [ Transfers of working cells]
[ 187] arn  63    ,gr    66               [ Storing of the 3 inputed numbers]
[ 188] arn  64    ,gr    67
[ 189] arn  65    ,gr    68
[ 190] hr   s1
[ 191] arn  66    ,gr    73               [ Storing of the 3 numbers produced by]
[ 192] arn  67    ,gr    74               [ passage of Produce random test number]
[ 193] arn  68    ,gr    75
[ 194] hv   49
[ 195] qq                                 [ Working cells]
[ 196] qq
[ 197] qq
[ 198] qq
[ 199] arn  73    ,gr    63               [ Results for later use as start numbers]
[ 200] arn  74    ,gr    64
[ 201] arn  75    ,gr    65
[ 202] hv   51

                                          [ Output of number of errors]
[ 203] pa   210          t4               [ Adjustment for 5 digits]
[ 204] dk   215                           [ Transform to machine number]
[ 205] ar   214   X                       [ Possibly round off, exchange M and R]
[ 206] mln  213                           [ Prepare ]
[ 207] tk   30    ,ga    209              [ output of 1 digit]
[ 208] sy   16    V LZ                    [ If digit=0 then write 0]
[ 209] sy                                 [ else write digit]
[ 210] bt   4            t-1              [ Counter for output of 5 digits]
[ 211] hv   206                           [ Jump within 5 digits]
[ 212] hr   s1                            [ else exit]

[ 213]  10                                [ Constants]
[ 214]   1
[ 215]  100000
[ 216] qq                                 [ 40 cells for the instruction sk 216]
```

```
i= 256                                  [ Simultaneous calculating by adder test 5]
[ 256] gr   309                         [ Only intended for corrections]
[ 257] arn  304   ,vy   16              [ Entry by start of adder test]
[ 258] gr           M                   [ Set address for floating-point overflow]
[ 259] pan  296   X      t1
[ 260] arfn 305
[ 261] grf  310
[ 262] arfn 306
[ 263] grf  311
[ 264] arfn 307
[ 265] grf  312                         [ Transfers of test numbers]
[ 266] arfn 308                         [ Adder test]
[ 267] grf  313
[ 268] arf  310
[ 269] mkf  311
[ 270] acf  312
[ 271] anf  313
[ 272] mlf  310   X
[ 273] scf  311
[ 274] srf  312
[ 275] mtf  313
[ 276] grf  310
[ 277] snf  311
[ 278] dkf  312
[ 279] gmf  313
[ 280] dlf  310
[ 281] ar   310
[ 282] mk   311
[ 283] ac   312
[ 284] an   313
[ 285] ml   310   X
[ 286] sc   311
[ 287] sr   312
[ 288] tl   -1
[ 289] hk   60        NKB               [ Jump to drum test if transfer is finished]
[ 290] mt   313
[ 291] gr   310
[ 292] sn   311
[ 293] dk   312
[ 294] gm   313
[ 295] dl   310
[ 296] bt   1               t1          [ Counter for 512 passages]
[ 297] hv   268                         [ Jump back within 512 passages]
[ 298] sr   309   ,ck                   [ else subtract result]
[ 299] sy   1        LZ                 [ Write ,1, if adder is ok]
[ 300] hv   259      LZ                 [ Jump to new test if adder is ok]
[ 301] zq            LKA                [ Stop if adder error\KA=1]
[ 302] sy   3                           [ Write ,3, if adder error]
[ 303] hv   259                         [ Jump to new test]
[ 304] hv   269
[ 305]  1234567890                      [ Test number]
[ 306]  2345678901
[ 307]  3456789012
[ 308]  4567890123
[ 309]  275/446/57/624                  [ Result of adder test]
[ 310]  0
[ 311]  0
[ 312]  0
[ 313]  0                               [ 40 cells for the instruction lk 314]
[ 314]  0
```

[Test Program No. 14. Reader Test. 15/1-1965.

The program is to be used with an infinite 8-hole punched paper tape containing sets of characters with odd parity and values ranging from 0-127. Between each set of 128 characters there is about 1 inch of blank tape. Such tapes can be produced with Test Program A 206.

Input of the test tape should be started with blank tape and the program stops in case of error. When the START button is pressed, the program proceeds with the next character.

In case of error the character read is to be found in the R register and the character which it ought to have been in the p register. The p register can be altered manually during error stop.]

```
i=41
zq    LKB              ; stop if KB
zq    NKB              ; stop if not KB
vy                     ; choose reader
pm  127  D             ; set mask for bits 3-9
pp  1023               ; set first character to -1
lyn r-4,   pp p+1      ; read, add 1 to p
cm  p    D             ; compare
zq                     ; error stop
hv  r-3                ; jump
e41
```

Should it be impossible to input the test program by means of the reader, it can be done manually from the console by insertion of the following in cells 0-3:

```
[0]  lyn  2,   pp p+1; bits No. 8,19-22,24,25,31-35,38-40
[1]  cm p D           ; bits No. 20,24,25,28,29,32
[2]  zq               ; bit No. 25
[3]  hv               ; bits No. 20-22
```

Also the following must be inserted in the registers:

```
by = 0
M  = 127x10-9; bits No. 3-9
p  = 1023    ; bits No. 0-9
```

The test is started by a jump to cell No. 0 and the test is performed as described above.

Rialto, 25. april 1968
JFM
t15

Test 15, Input and Output of 5 track paper tape

The program is used for test of input (from RC-2000) and output (to Facit Punch) of 5 track paper tape. The program is published as a binary paper tape, and may be loaded to the core store by the loaderprogram on track 0 for HJÆLP or HELP 3.

Subprograms

The program may be used in four different ways. The actual subprogram is selected when the program types:

Type c for copy, t for typeout, i for inputtest, o for outputtest:

The programs which may be selected are:

1. Copying of 5 track paper tape (type c)
2. Output of a special 5 track test tape (type o)
3. Input and test of the tape produced by program 2 (type i)
4. Conversion from 5 track paper tape in TELEX code to typewriter output in GIER code (type t)

The subprograms 1,2 and 4 may be stopped by setting KA=1. This causes a return to the startsituation and a new subprogram may be selected.

Test C-16

GIER-CP

Technical Center, 26.9.1968
JFM

The program is used for test of Mode 5 in GIER. Mode 5 is executed whenever the HP-button is pressed. The program is published as a B-C program only.

Method

After start (in Core[41]) the following happens:

1. Track 0 is saved in the core store during the test. Core[128:167]:=Track[0];
2. A special track 0 (part of the program) is written.
3. Track 38 is destroyed with special information.
4. The instructions

   [512] vy 0, vk 87

   [513] hv 512

   are moved to Core[512:513] and entered.

The program will now loop in 512 and 513 until the HP-button is pressed. When this happens the following test sequence is executed:

1. Core[0:39] is stored on track 38 (by Mode 5)
2. Track 0 is read to the core store (Core[0:39]:=Track[0]). (by Mode 5)
3. The instruction counter CT (r1) is stored in the address part of Core[0] and the remaining bits cleared. (by Mode 5);
4. Core[1] is enterdd in either rigth halfword or in left halfword. The entry depends on the state of the h-bit when HP was pressed. (by Mode 5);
5. Now the functions 1-4 above are checked by the program on track 0.
6. A jump to the instructions in Core[512:513] is performed and the test continue.

In case of errors, messages are given on the typewriter.

Use of KA,KB

KB=1	The program stops in Core[41] when the tape has been read.
KA=0	The normal state of KA during the test.
KA=1	After press on HP-button and the following test, the old track 0 is reestablished. One more press on HP will then activate the normal function of track 0 and the test is finished.

Error Messages

The following five messages are possible during the test, and the meaning is considered evident:

1. wrong tk stored
2. wrong r1 stored
3. -entry in Core[1]
4. Core[0:39] not stored on track 38
5. track 0 not transferred to core store

```
;slip<
;Test of HP function and mode 5, 1.8.68.4

b i=41, a40

[Core part of test,executed after track 0]

        zq              LKB         ; if kbon then STOP
        vy      512, grn     -1 ; set HP-inhibit; sum:=0;
        pa       a7   t     207 ; setsum of track 38:
        pa       a8   t      39 ;
a7:     arn     207   t       1 ;
        ar      1.8   DLA        ; include marks in sum
        ar      1.9   DLB        ;
        sc       -1              ; sum:=sum-R;
a8:     bt       39   t      -1 ;
        hv       a7              ;
        pm       -1, gm     228 ;
        vk        0   lk    128 ; save present track 0 in core;
        sk      168, vk     38 ; set new track zero;
        sk      208, lk      0
        sk      288, vk      0 ; noise to track 38 after setting core;
        pm       a9, arn    a10 ;
        gm      512   HA         ; set fixed program in core;
        gr      513   H          ; program used for interrupt by HP;
        hv      512              ; goto program;

a9:
[512]   vy        0  vk      87 ; release HP-inhibit; set tk to 87;
a10:
[513]   hv      512              ; goto Core[512];


a1:     gr      a12              ; saveR:=R; error in tk:
        hs      a11              ; writetext(<<
        qq      a13              ; wrong tk stored>);
        arn     a12, hr      s1 ; R:=saveR; return;

a2:     gr      a12              ; saveR:=R; error in CT:
        hs      a11              ; writetext(<<
        qq      a14              ; wrong r1 stored>);
        arn     a12, hr      s1 ; R:=saveR; return;

a3:     hs      a11              ; improper entry to track 0: writetext(<<
        qq      a15              ; -entry in Core[1]>);
        hr       s1              ; return;

a6:     hs      a11              ; errror track 38: writetext(<<
        qq      a16              ; Core[0:39] not stored on track 38>);
        hr       s1              ; return;

a13:    k64 ,58twrong tk stored;58.
a14:    k64 ,58twrong r1 stored;58.
a15:    k64 ,58t-entry in Core[1];58.
a16:    k64 ,58tCore[0:39] not stored on track 38;58.
```

```
a11:  [textprint of HELP 3]
      gs     r5, an      s ;
      is     s1  MA       ;
      ps    (s), arn   r-2 ;
      pm      s, ps     s1 ;
      cl     -6, ca    10.5 ;
      ps     -1  hr     s1 ; return;
      ca   15.5, hh    r-h ;
      tk     -h, ga     r2 ;
      ca     63, it      1 ;
      sy      0  hvm   r-5 ;

a12:  qq      0, qq      0 ; saveR;
```

d i=168

[New track 0 replacing the existing in the machine.
Handles and checks all actions taken by the HP-button
and mode 5]

```
[ 0]  qq      0   t    517 ; used for save OT;
      it    311, pt      0 ; normal entry: set bits(10-19) of Core[0];
      gk     -1, arn    -1 ; save by and tk; Rcount:=tk;
      vy    529, tk     10 ; set HP-inhibit;
      nc     87, hs     a1 ; if tk≠87 then call error;
      arn     0             ; R:=Core[0];
      ca    512, hv   ra17 ;
      ca    513  hv   ra17 ;
      ca      1, hv   ra17 ;
      ca      2, hv   ra17 ;
      ca      3, hv   ra17 ;
      hs     a2            ;
a17:  tk     10            ;
      nc    311, ca      0 ;
      hv   ra18            ;
      hs     a3            ;
a18:  vk     38, lk     288 ; read stored track 38 to core;
      vk     38, grn     -2 ; sum:=0;
      pa   ra4   t    287 ; setsum of track 38 in core[-2];
      pa   ra5   t     39 ;
a4:   arn  287   t      1 ;
      ar   1.8   DLA       ;
      ar   1.9   DLB       ;
[20]  sc     -2            ;
a5:   bt     39   t     -1 ;
      hv   ra4             ;
      arn    -2            ; R:=computed sum;
      hv   ra19   NZ       ; if R ≠ 0 then call error;
a20:
[27]  sk    208, lk      0 ; switch to new track 38;
[28]  hk     r1, hv      r ;
a19:  arn   289, nc    311 ;
      hs     a6            ;
      hv   ra20            ;
      qq                   ;
      qq                   ;
      qq                   ;
      qq                   ;
[34]  prn    1.3   DXIZA   ; Primitive input of HJELP:
      tl     -7, ly     r1 ;
      pi      0   IZA t508 ;
      xr      0   XIZB     ;
      hv     35   IZB      ;
[35]  hk     r1, VIO  t+1  ;
```

```
d i=208
[Core 0:39 during the test. Stored on track 38 after IP.
The storing is performed by mode 5.]

[ 0]  qq                        ;
      vy    529, sy     64 ; writetext(<<
      sy     58, sy     19 ; track 0 not transferred to core store>);
      sy     41, sy     49 ;
      sy     51, sy     34 ;
      sy      0  sy     16 ;
      sy      0  sy     37 ;
      sy     38, sy     19 ;
      sy      0  sy     10 ;
      sy     41, sy     49 ;
[10]  sy     37, sy     18 ;
      sy     54  sy     53 ;
       sy     41, sy      41 ;
      sy     53  sy     52 ;
      hv     26              ;
      qq                     ;
      qq                     ;
      qq                     ;
      qq                     ;
      qq                     ;
[20]  qq                     ; sumbalance for track;
      qq                     ;
      qq                     ;
      qq                     ;
      qq                     ;
      qq                     ;
      qq                     ;
      qq                     ;
[28]  hk     r1, hv      r  ; wait for transfer from drum;
      vk     38, sk    288  ; noise to track 38;
      hv    512   MXA       ; if - keen then exit to core;
      vk      0  sk    128  ; regenerate track 0 of machine;
[30]  vk      0  hv    512  ; exit to core;
      qq                     ;
[34]  prn   1.3   DXIZA      ; Primitive input of HJELP:
      tl     -7, ly     r1 ;
      pi      0   LZA t508 ;
      xr      0   XIZB     ;
      hv     35    LZB     ;
[39]  gr     41   IRC t-1 ;

s
e41
```

```
 25    1.8.68  849
start,image<
exit,10<
p26    1.8.68  e11
r<

1
294   41    248   300
a1     62
a2     66
a3     70
a4     188
a5     192
a6     73
a7     45
a8     49
a9     60
a10    61
a11    95
a12    105
a13    76
a14    79
a15    82
a16    87
a17    180
a18    184
a19    198
a20    196

 27    1.8.68  e42
binout,0,41..512<
print,p41..75,t76..94,p95..105,p168..247<
```

```
  41. zq                    LKB
  42. vy   512             ,grn  -1
  43. ps   45                    t207
  44. ps   49                    t39
  45. srn  207                   t1
  46. sr   2               D  LA
  47. sr   1               D  LB
  48. sc   -1
  49. bt   39                    t-1
  50. hv   45
  51. pm   -1              ,gm   228
  52. vk                   ,lk   128
  53. sk   168             ,vk   38
  54. sk   208             ,lk
  55. sk   288             ,vk
  56. pm   60              ,srn  61
  57. gm   512               MA
  58. gr   513               M
  59. hv   512
  60. vy                   ,vk   87
  61. hv   512
  62. gr   105
  63. hs   95
  64. qq   76
  65. srn  105             ,hr   s1
  66. gr   105
  67. hs   95
  68. qq   79
  69. srn  105             ,hr   s1
  70. hs   95
  71. qq   89
  72. hr   s1
  73. hs   95
  74. qq   87
  75. hr   s1

  76.
wron
  77. g tk s
  78. tored
  79.
wron
  80. g x1 s
  81. tored
  82.
-ent
  83. xy in
  84. Core
  85. [1]
  86.
  87.
Co
  88. re[0
  89. :39
  90. ] not
  91.  store
  92. d on t
  93. rack 3
  94. 8

  95. gs   x5 [100]      ,sn   s
  96. ig   s1                  MA
```

```
 98. pm   s              ,ps   s1
 99. cl   -6             ,ca   160
100. ps   -1             ,hx   s1
101. ca   2h0            ,hh   x-h [97]
102. tk   -h             ,ga   x2 [10h]
103. ca   63             ,it   1
10h. sy                  ,hvn  x-5 [99]
105. qq                  ,qq

168. qq                        t517
169. it   311            ,pt
170. gk   -1             ,exn  -1
171. vy   529            ,tk   10
172. nc   87             ,hs   62
173. exn
17h. ca   512            ,hv   x6 [180]
175. ca   513            ,hv   x5 [180]
176. ca   1              ,hv   x4 [180]
177. ca   2              ,hv   x3 [180]
178. ca   3              ,hv   x2 [180]
179. hs   66
180. tk   10
181. nc   311            ,ca
182. hv   x2 [18h]
183. hs   70
18h. vk   38             ,lk   288
185. vk   38             ,grn  -2
186. pa   x2 [188]             t287
187. pa   x5 [192]             t39
188. exn  287                  t1
189. ex   2              D  LA
190. ex   1              D  LB
191. sc   -2
192. bt   39                   t-1
193. hv   x-5 [188]
19h. exn  -2
195. hv   x3 [198]        MZ
196. gk   208            ,lk
197. hk   x1 [198]       ,hv   x [197]
198. exn  % c&9          ,nc   311
199. hs   73
200. hv   x-h [196]
201. qq
202. qq
203. qq
20h. qq
205. pvm  6h             XD IZA
206. tl   -7             ,ly   x1 [207]
207. pi                      IZA t508
208. qq
209. vy   529            ,sy   6h
210. sy   58             ,sy   19
211. sy   h1             ,sy   h9
212. sy   51             ,sy   3h
213. sy                  ,sy   16
21h. sy                  ,sy   37
215. sy   38             ,sy   19
216. sy                  ,sy   19
217. sy   h1             ,sy   h9
218. sy   37             ,sy   18
219. sy   5h             ,sy   53
220. sy   h1             ,sy   h1
221. sy   53             ,sy   52
222. hv   26
223. qq
22h. qq
```

```
227. qq
228. qq
229. qq
230. qq
231. qq
232. qq
233. qq
234. qq
235. qq
236. hk   x1 [237]      ,hv   x [236]
237. vk   38            ,sk   288
238. hv   512              NKA
239. vk                 ,sk   128
240. vk                 ,hv   512
241. qq
242. xmn  64            XD IZA
243. t1   -7            ,ly   x1 [244]
244. xi                    IZA t508
245. xx                 X  IZB
246. hv   35               LZB
247. gx   h1               NRC t-1
```

Rialto, 16 april 1968.
JFM
t 17

Test 17, Printer

The program is intended for test of the lineprinter. It is published as a binary papertape and may be loaded to the core store by the primitive input program on track 0 of either HJÆLP or HELP 3.

Method

The program prints characters in 159 positions on the printer.
The characters printed depends on the setting of KA and KB in the following manner:

1) KA=0, KB=0 : One line is printed consisting of one specific character. This character is typed on the typewriter when the program types Hej .

2) KA=1 , KB=0 : Each line contains all characters in both cases. The printing continues as long as KA is one. The first character in each line will be the one typed in the Hej-situation.

3) KA=0 , KB=1 : Each line contains the same character, but the both in upper case and lower case. The printerposition for shift between UC and LC will be shifted one towards left for each line. The first line will consist of the character typed in the Hej-situation.

4) KA=1 , KB=1 : Each line contain all characters and the printing is shifted one position towards left for each new line.

Blind symbols

All blind characters are printed as SPACE. This is also valid for STOPCODE and TABULATOR.

RC-RIALTO
21.3.1966
TESTPROGRAM B-18
JFM

The program is intended for test of disk files, in the cases where the disk unit replaces the usual drum unit; of course the program can be used for drum test, too.

PRINCIP

After reading in the program tape, the user specifies the area of the disk file he want to test. Then the program write known information in each word of this area. The information consist of word of the following format:

qq 'disknumber'. 9 + 'group'. 19 + 'sector'. 29 + 'wordnumber'. 39

After writing, this information is read to the core store in groups of eight sectors and compared with the correct information. This correct information is generated during reading and adjusted according to the current value of disk, group, sector, and word.

STORAGE LAYOUT

The core store is used as follows:

```
  0 -  39 : track 0 of HELP
       41 : the instruction zq LKB
  42 - 143 : program administration
 144 - 244 : number read routine
 245 - 274 : number print routine
 275 - 289 : text print routine
 290 - 316 : routine for generation of correct information
 317 - 334 : error print routine
       335 : last group + 1
       336 : last sector + 1
       337 : disk unit number
       338 : first group
       339 : last group
       340 : first sector
       341 : last sector
       342 : error used
       343 : current value of group
       344 : current value of sector
       345 : address for choice of disk and tensgroup
       346 : not used
       347 : 4 (Floating)
       348 : 10     -
       349 : 961    -
       350 : -40 (address)
       351 : 960      -
 352 - 382 : text strings
 383 - 702 : 8 sectorsoof correct information
 703 -1022 : 8 sectors read from the disk
```

RUNNING

After reading in the program types:

Disk No.:	0	(type disk unit number)
First Group:	0	(type the wanted start group)
Last Group:	0	(- - - end -)
First Sector:	0	(- - - start sector)
Last Sector:	319	(- - - end -)

First sector must be less than last sector and both < 960. First sector will be rounded to the nearest lower number divisible by 8. Last sector will be rounded to the nearest higher number divisible by 7, thus giving the possibility of writing and testing one entire track of eight sectors in one loop. First group must be less than or equal to last group and both less than 10. Disk unit must be less than 4. After typing last section the program starts. When one complete testcycle is finished, the program jumps to the start situation, if KB = 1 and stops in cell 41. If KB = 0, the program goes through another complete testcycle.

NUMBERS

The numbers to be typed must conform with the syntax for input in Gier-Algol III, because this is the input routine which is used.

ERRORS

When an error has been found, two lines are printed. One line in black showing the written information and one line in red showing the information which was read from the disk. The first error gives printing of a heading:

Disk Group Sect. Word

and the error print could be:

0	0	25	17	(black)
0	0	875	19	(red)

Rialto, 18. march 1968
JFM
t19

Test 19, Drum - Disk

The program is used for test of the drum(s) or the disk file replacing the drum(s).

The program is published as a binary tape to be read by the primitive input program on track 0 of HELP 3 or of HJÆLP. The program starts in cell 41 with a ZQ instruction. There exist five versions of the program:

1) For one drum
2) For two drums
3) For three drums
4) For Anelex model 80 Disk File
5) For CDC 9433 Disk File

Method

The program writes a bitpattern on the entire drumarea, reads this back and checks it. The bitpatterns used are the same as used in test 10, test 11 and buffertest 3. After one cycle the next pattern is written and so on, until all 84 different patterns has been tested. The program jumps to core 41 and stops. If one pushes START the entire cycle will be repeated.

Testinformation

The patterns used as testinformation are taken as 42 consecutive bits from the following larger pattern:

$$\underbrace{1,1,1,\ldots,1}_{42\text{ bits}},\underbrace{0,0,0,0,0,\ldots,}_{42\text{ bits}},\underbrace{1,1,1,\ldots,1}_{41\text{ bits}}$$

This means that a complete run consists of 84 cycles each writing, reading and testing one specific bitpattern.

Group selection

In three drum versions the actual group number may be typed as an integer $0 \leq n \leq 19$.

Use of indicator

The indicator is used as follows during the run:

OA: is one if an error has been detected.

PA: is set to one if printing of bitpattern is wanted when errors are detected. Is set to zero from the program when a new major cycle starts.

PB: is normally one. If PB is zero the printing of cylinder and head number in case of error will be omitted.

QA, QB: used internal in testcycles.

RA, RB: used internal in testcycles.

KA: If KA is zero the entire drumarea will be tested including track 0 of group 0. In this case tracks 0-31 should be open for writing. If KA is one tracks 0-31 of group zero are not tested. If one tests the entire area, track 0 will be saved in the core during the test and written back after the complete cycle. If one stops the program earlier, track zero may be reestablished by a manual jump to core [50] . (in 3 drum versions to core [52])

Erroroutput

In case of error the following information is typed on the console typewriter as one line:

1) In drummode of operation:

t ‹ selected track › w ‹ wordnumber › i m

2) In disk mode of operation:

g ‹ selected group› s ‹selected sector› w ‹wordnumber› i c ‹cylinder› h ‹head ›m

‹ i › or ‹m › means that an ‹ i › is typed when the error was in bits 0-39, and a ‹m › is typed if the error was in bits 40-41.

‹ m › in diskmode means that the m in case of error in bits 40-41 will be printed after the headnumber if PB is one.

If PA is one the corresponding bitpatterns are printed as two lines:

‹ bitpattern from table › (in black)

‹ bitpattern from drum or disk › (in red)

If an error occurs both in bits 0-39 and 40-41 the corresponding m will be printed after the last bit in the second line.

Rialto 19. march 1968
JFM
t20

Test 20 for Drum/Disk

The program is used for test of either the drum(s) or the diskfile replacing the drum. There exists the following versions of the program:

1) For Gier with one drum
2) For Gier with two drums
3) For Gier with three drums
4) For Gier with Anelex Diskfile
5) For Gier with CDC-9433 Diskfile

The programs is published as a binary papertape which may be read to the core store by the primitive input program on track 0 of either HELP-3 or HJÆLP.

Method

The program writes random information on random selected tracks (sectors). The sequence of random tracks (sectors) is generated from 3 typed startrandoms. The number of tracks to be tested in one programcycle are also typed. The pseudorandom information generated for a given track (sector) is based on the 3 randoms:

1. qq ⟨ current group ⟩ .9
2. qq ⟨ current track or sector ⟩ .9
3. ⟨ part of a textstring, a constant ⟩

Operating

After loading the program types:

Type 3 random integers:	(do this)
Number of tracks:	(type wanted value)

The number of tracks to be typed defines how many (not necessarily different) tracks one wishes to test in one majorcycle of the program.

After typing of startparameters the writing of information begins. When the writing is finished the reading and checking begins. When this is finished the program writes:

ok (When no errors have been found)

or - (If one or more errors have been detected).

After this message the program jumps the startsituation if KA = 1. KA = 0 gives a new run with the same number of tracks (sectors), but with another sequence of tracks.

KB

Normally one should have KB = 0. If KB = 1 the program types the track-number (or in case of diskfile the sector and groupnumber) whenever a track is selected for reading or writing. This feature may be useful if something happens in the selection of tracks.

Error printout

In case of error during read, the following information is typed:

A. Drumversions

t < drumtrack > w < word on track >
< bitpattern of written information in black >
< bitpattern of read information in red >

B. Diskversions

g < group > s < sector > w < word in sector >
< bitpattern of written information in black >
< bitpattern of read information in red >

; slip<

[Diagnostic program t 21 test of mode 1 and 4 oct.68 PEH

The program tests all possible combinations of the conditions in mode 1 and 4. If an error is detected output is given on typewriter according to the following table

output	type of non-successful function
g0, g1, g2	basic operation fullword, LH, RH halfword
a0, a1, a2	adress part of - - - - - -
n0, n1, n2	n-mark (S-mark) - - - - - -
i0, i1, i2	indirect adressing - - - - -
r0, r1, r2	relative-mark - - - - - -
s0, s1, s2	subroutinemark - - - - - -
p0, p1, p2	p-index - - - - - -
B	indicator condition
D	D-mark
f	floating-point mark
H	halfword mark added
h	- - - deleted
IK	IKC operation
K	K-part of indicator (bit pos.37)
T	T-part of indicator (bit pos.35)
Z	Z-part of indicator (bit pos.36)
L	LA or LB operation (bit pos.33, 34, 38, 39)
X	X-modification
V	V-modification
q	incremental operation
qs	incremental operation is erroneously treated as static or vice-versa

The letters may be combined, thus the message

isB

means that a mode 1 or 4 error is detected during execution of an instructio which is indirect adressed, s-marked and furnished with some indicator condition.

If KA is set during run the error message will be followed by a three-digit number which gives the adress of the cell from which the alarm was called.

Pressing of KB causes the machine to stop.

If no errors are detected an ok message is typed every 20 seconds.

Input of program.

The program may be read in like any other B or C test. However, if this does not work the program may be read in using as simple instructions as possible in this way:

1. RESET GIER.
2. place the tape in the reader starting in the middle of the 100 spaces found a few inches after the starting point used for old track 0.
3. clear the entire core store using hardware test 1.
4. clear R, M and by.
5. insert in cell 41 - 43 the following instructions:

41.	ly 43	bit pos. 4, 6,8, 9, 20, 21, 22, 24, 25
42	cl 76	- - 3, 6, 7, 21, 23
43.	gm	- - 21, 22, 24

6. start GIER in cell 41

When the program has been read in the primitive input program in cell 34 - 39 is re-established.

```
      i=27

[  27]      pm i+5      ;
[  28]      gm -2, M    ;
[  29]      pm i+4      ;
[  30]      gm 0   M    ;
[  31]      hv 40       ;
[  32]      ar a1       ;
[  33]      tk 42       ;
[  34]      pmm64XDIZA; primitive input
[  35]      tl -7,lyr1;
[  36]      pi IZAt508;
[  37]      xr X IZB    ;
[  38]      hv 35 LZB   ;
[  39]      gr41MRCt-1;
[  40]      tk 42       ;
[  41]      qq          ;
[  42]      vy 16       ;
[  43]      sy 29       ; red ribbon
[  44]      sy 58       ; LC
[  45]      sy 64       ;
[  46]      hv a0       ; goto start


;     alarm print

[  47]b0:   qq          ; work
[  48]      qq          ; constant
[  49]      sy 55       ; g
[  50]      qq          ;
[  51]      sy 49       ; a
[  52]      hv b1       ;
[  53]      sy 37       ; n
[  54]      hv b1       ;
[  55]      sy 57       ; i
[  56]      hv b1       ;
[  57]      sy 41       ; r
[  58]      hv b1       ;
[  59]      sy 18       ; s
[  60]      hv b1       ;
[  61]      sy 39       ; p
[  62]b1:   sy 16       ; [full-word]
[  63]      hv a2       ; goto s-print
[  64]      sy 55       ; g
[  65]      hv b2       ;
[  66]      sy 49       ; a
[  67]      hv b2       ;
[  68]      sy 37       ; n
[  69]      hv b2       ;
[  70]      sy 57       ; i
[  71]      hv b2       ;
[  72]      sy 41       ; r
[  73]      hv b2       ;
[  74]      sy 18       ; s
[  75]      hv b2       ;
[  76]      sy 39       ; p
[  77]b2:   sy  1       ; 1 [LH half-word]
[  78]      hv a2       ; goto s-print
[  79]      sy 55       ; g
```

```
[  80]      hv b3      ;
[  81]      sy 49      ; a
[  82]      hv b3      ;
[  83]      sy 37      ; n
[  84]      hv b3      ;
[  85]      sy 57      ; i
[  86]      hv b3      ;
[  87]      sy 41      ; r
[  88]      hv b3      ;
[  89]      sy 18      ; s
[  90]      hv b3      ;
[  91]      sy 39      ; p
[  92]b3:   sy  2      ; 2 [RH half-word]
[  93]      hv a2      ; goto s-print
[  94]      sy 60      ; B [indicator condition]
[  95]      sy 50      ;
[  96]      hv a2      ;
[  97]      sy 60      ; D
[  98]      sy 52      ;
[  99]      hv a2      ;
[ 100]      sy 54      ; f [floating-point mark]
[ 101]      hv a2      ;
[ 102]b4:   sy 60      ; H
[ 103]      sy 56      ; h
[ 104]      hv a2      ;
[ 105]      sy 60      ; IK [IKC operation]
[ 106]      sy 57      ;
[ 107]      sy 60      ; K
[ 108]      sy 34      ;
[ 109]      hv a2      ;
[ 110]      sy 60      ; L
[ 111]      sy 35      ;
[ 112]      hv a2      ;
[ 113]b5:   sy 40      ; q
[ 114]      hv a2      ;
[ 115]      sy 40      ; qs
[ 116]      sy 18      ; [static op treated as incremental
[ 117]      hv a2      ; or vice-versa]
[ 118]      sy 60      ; T
[ 119]      sy 19      ;
[ 120]      hv a2      ;
[ 121]      sy 60      ; X
[ 122]      sy 23      ;
[ 123]      hv a2      ;
[ 124]b6:   sy 60      ; XV
[ 125]      sy 23      ;
[ 126]      sy 60      ; V
[ 127]      sy 21      ;
[ 128]      hv a2      ;
[ 129]      sy 60      ; Z
[ 130]      sy 25      ;


;     s-print

[ 131]a2:   sy 58      ;
[ 132]      hv a3 NKA  ;
[ 133]      sy  0      ;
[ 134]      tk 42      ;
[ 135]      ar c2      ;
[ 136]      gr c3      ;
[ 137]      tl 80      ; R:=M:=0
[ 138]      gs 1+1     ;
```

```
[ 139]      qq [s]     ;
[ 140]      ar i-1     ;
[ 141]      cl 10      ;
[ 142]      ml c       ;
[ 143]c4:   ck -10     ;
[ 144]      ca  0      ;
[ 145]      hv i+2     ;
[ 146]      hv i+3     ;
[ 147]      arn i+1    ;
[ 148]      qq 16      ;
[ 149]      ga i+1     ;
[ 150]      sy         ;
[ 151]      tk 42      ;
[ 152]      ar c3      ;
[ 153]      ar a1      ;
[ 154]      gr c3      ;
[ 155]      ca  0      ;
[ 156]      hv i+3     ;
[ 157]      ac c6      ;
[ 158]      hv a3      ;
[ 159]      ml c1      ;
[ 160]      hv c4      ;
[ 161]c0:   m 10-2     ;
[ 162]c1:   qq 10.39   ;
[ 163]c2:   qq 2.39    ;
[ 164]c3:   qq         ; work

;    message counter

[ 165]a3:   sy  0      ;
[ 166]      tk 42      ;
[ 167]      ar c6      ;
[ 168]      ar a1      ;
[ 169]      gr c6      ;
[ 170]      ca  0      ;
[ 171]      hv a0      ;
[ 172]      sy 64      ;
[ 173]      ar c5      ;
[ 174]      gr c6      ;
[ 175]      hv a0      ;
[ 176]c5:   qq 20.39   ;
[ 177]c6:   qq 19.39   ;

;     reset and start

[ 178]a0:   tk 42      ;
[ 179]      ar c7      ;
[ 180]      gr c8      ; reset ok-counter
[ 181]      pp  0      ;
[ 182]      ps  0      ;
[ 183]      hv i+2     ;
[ 184]a1:   qq -1.39   ; constant
[ 185]      tk 42      ;
[ 186]      hv i+2 NKB;
[ 187]      zq  LKB    ;
[ 188]      ud -2      ; test of adress, fullword
[ 189]      sr a1      ;
[ 190]      ca  0      ;
[ 191]      hv i+2     ;
[ 192]      hs 2b      ; print(ga0)
[ 193]      qqtb4XVD   ; [qq  ,hv b4], test of +half-word mark, print(H)
```

```
[ 194]      qq          ; dummy
[ 195]a4:   ar a1       ; test of LA, LB
[ 196]      hs 8b4 LA   ; print(L)
[ 197]      hs 8b4 LB   ; print(L)
[ 198]      pi -1       ; test of T, Z, K
[ 199]      hs 5b5NTC   ; print(T)
[ 200]      hs 5b6 LZ   ; print(Z)
[ 201]      tk 42       ;
[ 202]      hs 5b6 NZ   ; print(Z)
[ 203]      hs 5b4 LO   ; print(K)
[ 204]      tk 42       ; R:=0
[ 205]a5:   pm a1       ; test of X, M:=-1.39
[ 206]      cl 40  X    ;
[ 207]      hv i+2 LZ   ;
[ 208]      hs 8b5 NZ   ; print(X)
[ 209]      pm a1  X    ; R:=-1.39, M:=0
[ 210]      sr a1       ; R:=0
[ 211]      hv i+2 LZ   ;
[ 212]      hs 8b5 NZ   ; print(X)
[ 213]      tk 42  X    ; M:=0
[ 214]      xr          ; R:=0
[ 215]a6:   hv i+2 LZ   ;
[ 216]      hs 8b5 NZ   ; print(X)
[ 217]      ar a1  X    ; M:=-1.39
[ 218]      xr          ; R:=-1.39
[ 219]      sr a1       ; R:=o
[ 220]      hv i+2 LZ   ;
[ 221]      hs 8b5 NZ   ; print(X)
[ 222]      qq     V    ; test of V
[ 223]      hs 2b6      ; print(V)
[ 224]      pm a1  XV   ; test of XV
[ 225]a7:   hs b6       ; print(XV)
[ 226]      sr a1       ;
[ 227]      hv i+2 LZ   ;
[ 228]      hs b6  NZ   ; print(XV)
[ 229]      arf a1  X   ; test of floating-point mark
[ 230]      hv i+2 NZ   ;
[ 231]      hs 8b3      ; print(f)
[ 232]      tk 42       ;
[ 233]      ar a1  D    ; test of D
[ 234]      ca a1       ;
[ 235]      hv i+2      ;
[ 236]      hs 5b3      ; print(D)
[ 237]      tk 42       ;
[ 238]      ar a1       ; test of n, full-word
[ 239]a8:   qqn         ;
[ 240]      hv i+2 LZ   ;
[ 241]      hs 6b  NZ   ; print(n0)
[ 242]      ca (1b)     ; test of i, full-word
[ 243]      hv i+2      ;
[ 244]      hs 8b       ; print(i0)
[ 245]      tk 42       ;
[ 246]      ar ra1      ; test of r, full-word
[ 247]      sr a1       ;
[ 248]      hv i+2 LZ   ;
[ 249]a9:   hs 10b NZ   ; print(r0)
[ 250]      ps a1       ; test of s, full-word
[ 251]      ar s        ;
[ 252]      sr a1       ;
[ 253]      ps  0       ;
[ 254]      hv i+2 LZ   ;
[ 255]      hs 12b NZ   ; print(s0)
[ 256]      pp a1       ; test of p, full-word
```

```
[ 257]      ar  p      ;
[ 258]      sr a1      ;
[ 259]a10:  pp  0      ;
[ 260]      hv i+2 LZ  ;
[ 261]      hs 14b NZ  ; print(p0)
[ 262]      ar a1      ; test of LH hailf-word
[ 263]      ps 42      ;
[ 264]      ud   ,     ; LH, basic op
[ 265]      ps  0      ;
[ 266]      hv i+2 LZ  ;
[ 267]      hs 2b1 NZ  ; print(g1)
[ 268]      ps -1      ; test of LH adress
[ 269]a11:  ud -2,     ;
[ 270]      ps 0       ;
[ 271]      sr a1      ;
[ 272]      hv i+2 LZ  ;
[ 273]      hs 4b1 NZ  ; print(a1)
[ 274]      ar a1      ; n, LH
[ 275]      qqn  ,     ;
[ 276]      hv i+2 LZ  ;
[ 277]      hs 6b1 NZ  ; print(n1)
[ 278]      pi i-4     ;
[ 279]      gi i+1     ;
[ 280]      ar (i-6),  ; i, LH
[ 281]a12:  sr a1      ;
[ 282]      hv i+2 LZ  ;
[ 283]      hs 8b1 NZ  ; print(i1)
[ 284]      pi a1-i-2  ;
[ 285]      gi i+1     ;
[ 286]      ar ra1,    ; r, LH
[ 287]      sr a1      ;
[ 288]      hv i+2 LZ  ;
[ 289]      hs 10b1NZ  ; print(r1)
[ 290]      ps a1      ;
[ 291]      ar  s,     ;
[ 292]      sr a1      ;
[ 293]a13:  ps. 0      ;
[ 294]      hv i+2 LZ  ;
[ 295]      hs 12b1NZ  ; print(s1)
[ 296]      pp a1      ;
[ 297]      ar  p,     ;
[ 298]      sr a1      ;
[ 299]      pp  0      ;
[ 300]      hv i+2 LZ  ;
[ 301]      hs 14b1NZ  ; print(p1)
[ 302]      ,tl [VM]   ; test of -half-word
[ 303]a14:  hv i+2     ;
[ 304]      hs 1b4     ; print(h)
[ 305]      tk 42      ;
[ 306]      ar a1      ; test of RH half-word
[ 307]      ps 42      ;
[ 308]      ,ud        ; basic op
[ 309]      qq         ; dummy
[ 310]      ps  0      ;
[ 311]      hv i+2 LZ  ;
[ 312]      hs 2b2 NZ  ; print(g2)
[ 313]a15:  ps -1      ; RH adress
[ 314]      ,ud -2     ;
[ 315]      sr a1      ;
[ 316]      ps  0      ;
[ 317]      hv i+2 LZ  ;
[ 318]      hs 4b2 NZ  ; print(a2)
[ 319]      ar a1      ; n, RH
```

```
[ 320]      ,qqn       ;
[ 321]      hv i+2 LZ  ;
[ 322]      hs 6b2 NZ  ; print(n2)
[ 323]a16:  ,ca (1b)   ; i, RH
[ 324]      hv i+2     ;
[ 325]      hs 8b2     ; print(i2)
[ 326]      ,ar ra1    ; r, RH
[ 327]      sr a1      ;
[ 328]      hv i+2 LZ  ;
[ 329]      hs 10b2NZ  ; print(r2)
[ 330]      ps a1      ; s, RH
[ 331]      ,ar  s     ;
[ 332]      sr a1      ;
[ 333]a17:  ps  0      ;
[ 334]      hv i+2 LZ  ;
[ 335]      hs 12b2NZ  ; print(s2)
[ 336]      pp a1      ; p, RH
[ 337]      ,ar  p     ;
[ 338]      sr a1      ;
[ 339]      pp  0      ;
[ 340]      hv i+2 LZ  ;
[ 341]      hs 14b2NZ  ; print(p2)
[ 342]      ,tl  [VM]  ; test of -half-word mark
[ 343]a18:  hv i+2     ;
[ 344]      hs 1b4     ; print(h)
[ 345]      pa i+1     ; test of increment
[ 346]      qq  ,qq 1  ;
[ 347]      ar i-1     ;
[ 348]      ca  0      ;
[ 349]      hv i+2     ;
[ 350]      hs b5      ; print(q)
[ 351]      tk 42      ;
[ 352]      pa i+1     ; test of -,B∧q
[ 353]a19:  ar ta1 NZ  ;
[ 354]      hv i+3 LZ  ;
[ 355]      ud b5  NZ  ; print(qB)
[ 356]      hs 2b3 NZ  ;
[ 357]      ar a1      ; test of q
[ 358]      pa i+1     ;
[ 359]      sr  ta1    ;
[ 360]      hv i+2 LZ  ;
[ 361]      hs b5  NZ  ; print(q)
[ 362]      qq tb4 XVM; [qq  ,hs b4]
[ 363]a20:  qq         ; test of +half-word mark, print(H)
[ 364]      tk 42      ;
[ 365]      ca (1b)    ; test of i, full-word
[ 366]      hv i+2     ;
[ 367]      hs 8b      ; print(i0)
[ 368]      tk 42      ; test of increment
[ 369]      ga i+12    ; decoding of TD≠0
[ 370]      ar a1      ;
[ 371]      qq(i+10)t512
[ 372]      qq(i+9)t256
[ 373]      qq(i+8)t128
[ 374]a21:  qq(i+7)t64;
[ 375]      qq(i+6)t32;
[ 376]      qq(i+5)t16;
[ 377]      qq(i+4)t8  ;
[ 378]      qq(i+3)t4  ;
[ 379]      qq(i+2)t2  ;
[ 380]      qq(i+1)t1  ;
[ 381]      ca -1      ;
[ 382]      hv i+2     ;
```

```
[ 383]      hs b5      ; print(q)
[ 384]a22:  ga i+1     ; decoding of
[ 385]      pi -1 t1   ; static op
[ 386]      tk 42      ;
[ 387]      ar i-2     ;
[ 388]      ca -1      ;
[ 389]      hv i+2     ;
[ 390]      hs 2b5     ; print(qs)
[ 391]      pi -500    ;
[ 392]      gi i+1     ; decoding of non-static op
[ 393]      bt-500t-100
[ 394]a23:  hv i+2     ;
[ 395]      hs 2b5     ; print(qs)
[ 396]      tk 42      ;
[ 397]      pa i+1     ; decoding of
[ 398]      nc 0 t-1   ; non-static op
[ 399]      hv i+2     ;
[ 400]      hs 2b5     ; print(qs)
[ 401]      qq tb4 XVM; test of +h
[ 402]      qq         ;
[ 403]      tk 42      ;
[ 404]      pa i+2     ; test of step 12
[ 405]      nt i+1     ; increment^r
[ 406]a24:  ca  r      ;
[ 407]      hv i+3     ;
[ 408]      ud 10b     ; print(rq)
[ 409]      hs b5      ;
[ 410]      pa i+3     ;
[ 411]      ps i+2     ;
[ 412]      nt i+1     ; increment^s
[ 413]      ca   s     ;
[ 414]      hv i+4     ;
[ 415]      ps  0      ;
[ 416]      ud 12b     ; print(sq)
[ 417]a25:  hs b5      ;
[ 418]      ps  0      ;
[ 419]      pa i+3     ;
[ 420]      pp i+2     ; increment^p
[ 421]      nt i+1     ;
[ 422]      ca  p      ;
[ 423]      hv i+3     ;
[ 424]      ud 14b     ; print(pq)
[ 425]      hs b5      ;
[ 426]      pp  0      ;
[ 427]      tk 42      ;
[ 428]      hv 1a1 NZ  ;
[ 429]a26:  ar a1      ;
[ 430]      ncn   LZ   ; test of -,B^n
[ 431]      hv i+3     ;
[ 432]      ud 6b      ; print(nB)
[ 433]      hs 2b3     ;
[ 434]      tk 42      ;
[ 435]      ar a1      ; test of -,B^(s)
[ 436]      ga i+1     ;
[ 437]      qq         ;
[ 438]      ps i-1     ;
[ 439]      nc (s) LZ  ;
[ 440]      hv i+5     ;
[ 441]a27:  ps  0      ;
[ 442]      ud 8b      ; print(isB)
[ 443]      ud 12b     ;
[ 444]      hs 2b3     ;
[ 445]      ga i+1     ; test of (r) and (s)
```

```
[ 446]      qq          ;
[ 447]      ps i-1      ;
[ 448]      ca (s)      ;
[ 449]      hv i+4      ;
[ 450]      ps  0       ;
[ 451]a28:  ud 8b       ; print(is0)
[ 452]      hs 12b      ;
[ 453]      ps  0       ;
[ 454]      ca (r-8)    ;
[ 455]      hv i+3      ;
[ 456]      ud 8b       ; print(ir0)
[ 457]      hs 10b      ;
[ 458]a29:  pp -1       ; test of IK (mode 4)
[ 459]      pi  0       ;
[ 460]      tk 42       ;
[ 461]      nc  0 IKC   ; p:=0, in:=-1
[ 462]      hs b4       ; if +h then print(H)
[ 463]      gp i+1      ;
[ 464]      ca          ;
[ 465]      hv i+2      ;
[ 466]a30:  hs 3b4      ; print(IK)
[ 467]      gi i+1      ;
[ 468]      qq          ;
[ 469]      nc  0 IKC   ; p:=-1, in:=0
[ 470]      hs b4       ; if +h then print(H)
[ 471]      ar i-3      ;
[ 472]      ca -1       ;
[ 473]      hv i+2      ;
[ 474]      hs 3b4      ; print(IK)
[ 475]      gp i+1      ;
[ 476]      ca          ;
[ 477]a31:  hv i+2      ;
[ 478]      hs 3b4      ; print(IK)
[ 479]      gi i+2      ;
[ 480]      tk 42       ;
[ 481]      ca          ;
[ 482]      hv i+2      ;
[ 483]      hs 3b4      ; print(IK)
[ 484]      qq    V     ; test of V
[ 485]      hs 2b6      ; print(V)
[ 486]      pp -1       ; test of VIK
[ 487]      pi  0 VIKC  ; p:=0, in:=-1
[ 488]      hs i+2      ; print(VIK)
[ 489]a32:  hv i+4      ;
[ 490]      ud 2b6      ;
[ 491]      ud 3b6      ;
[ 492]      hv 3b4      ;
[ 493]      tk 42       ;
[ 494]      gp i+1      ;
[ 495]      ca          ;
[ 496]      hv i+2      ;
[ 497]      hs i-7      ; print(VIK)
[ 498]      hv 1a1 NZ   ; skip if +h
[ 499]      pm a1  X    ; test of X
[ 500]      hs 8b5 LZ   ; print(X)
[ 501]      pm a1 XIKC  ; test of XIK
[ 502]      gp i+1      ; R:=-1, M:=0, p:=-1, in:=0
[ 503]a33:  ca          ;
[ 504]      hv i+4 NZ   ;
[ 505]      ud 8b5      ; print (XIK)
[ 506]      ud 9b5      ;
[ 507]      hs 3b4      ;
[ 508]c7:   qq 2000.39  ; Ok counter
```

```
[ 509]c8:   qq          ; work
[ 510]      tk 42       ;
[ 511]      ar i-2      ;
[ 512]      ar a1       ;
[ 513]a34:  gr i-4      ;
[ 514]      hv 1a1 NZ   ; if 2000 successful runs then
[ 515]      sy 62       ; print(ok)
[ 516]      sy 38       ;
[ 517]      sy 34       ;
[ 518]      sy 29       ;
[ 519]a35:  hv a3       ; goto start and reset
```

e27

Rialto, 25 april 1968
JFM
t26

Test 26, Input/Output Typewriter

The program is used for test of the typewriter, when the typewriter is used as outputmedium. For test of inputfunctions one should use t9. The program is published as a binary punched papertape, which may be read to the core store by means of the primitive loaderprogram on track 0 of HJÆLP or HELP 3.

Method

The program writes lines as shown on the figure. These lines are constructed in such a way that all characters are typed and all functions (caseshift, carriage return, tabulation and ribbon shift) are activated several times. When the output on the figure has been typed the program repeats the entire process.

```
  123  456  789  0-<  qwe  rty  uio  påa
  ∨×/  =;[  ]()  ∧+>  QWE  RTY  UIO  PÅA
  123  456  789  0-<  qwe  rty  uio  påa
  ∨×/  =;[  ]()  ∧+>  QWE  RTY  UIO  PÅA
  123  456  789  0-<  qwe  rty  uio  påa
  ∨×/  =;[  ]()  ∧+>  QWE  RTY  UIO  PÅA
  123  456  789  0-<  qwe  rty  uio  påa
  ∨×/  =;[  ]()  ∧+>  QWE  RTY  UIO  PÅA
sdfg       hjkl       æøzx       cvbn       m,.-
SDFG       HJKL       ÆØZX       CVBN       M10:+
sdfg       hjkl       æøzx       cvbn       m,.-
SDFG       HJKL       ÆØZX       CVBN       M10:+
sdfg       hjkl       æøzx       cvbn       m,.-
SDFG       HJKL       ÆØZX       CVBN       M10:+
sdfg       hjkl       æøzx       cvbn       m,.-
SDFG       HJKL       ÆØZX       CVBN       M10:+
1234567890<  qwertyuiopå  asdfghjklæø  zxcvbnm,.-
∨×/=;[]()∧>  QWERTYUIOPÅ  ASDFGHJKLÆØ  ZXCVBNM10:+
1234567890<  qwertyuiopå  asdfghjklæø  zxcvbnm,.-
∨×/=;[]()∧>  QWERTYUIOPÅ  ASDFGHJKLÆØ  ZXCVBNM10:+
1234567890<  qwertyuiopå  asdfghjklæø  zxcvbnm,.-
∨×/=;[]()∧>  QWERTYUIOPÅ  ASDFGHJKLÆØ  ZXCVBNM10:+
1234567890<  qwertyuiopå  asdfghjklæø  zxcvbnm,.-
∨×/=;[]()∧>  QWERTYUIOPÅ  ASDFGHJKLÆØ  ZXCVBNM10:+
```